09/745,480

012291886 **Image available**
WPI Acc No: 1999-097992/199909
XRPX Acc No: N99-071359

Bus manager and control system for control of image input unit - in which bus arbitrator grants bus masters access to use of bus in accordance with stored conditions for starting and ending granting of bus use privilege

Patent Assignee: CANON KK (CANO)
Inventor: DATE A; FUJIWARA T; KATO K; MAEDA T; YOKOYAMA N

Number of Countries: 026 Number of Patents: 002
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 893766	A2	19990127	EP 98305792	A	19980721	199909	B
JP 11045225	A	19990216	JP 97200570	A	19970725	199917	

Priority Applications (No Type Date): JP 97200570 A 19970725
Cited Patents: No-SR.Pub
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 893766	A2	E	98	G06F-013/362	
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI					
JP 11045225	A		60	G06F-013/362	

Abstract (Basic): EP 893766 A

The bus manager includes a number of bus masters for connection to at least one bus, and a memory unit for storing start and end conditions for granting bus use privileges to each of the bus masters.

A bus arbitration unit grants the bus masters the 'bus use privilege' or deprives them of the 'bus use privilege' in accordance with the conditions, if there are bus use requests from the bus masters.

USE - Control of image input or output unit eg. scanner or printer.

ADVANTAGE - Overall processing speed of large quantities of data eg. image data, is increased without requiring software for each processing operation.

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平11－45225

(43)公開日　平成11年(1999)２月16日

(51)Int.Cl.[6]		識別記号	FI		
G06F	13/362	510	G06F	13/362	510D
	13/16	520		13/16	520C
	13/36	530		13/36	530B

審査請求　未請求　請求項の数26　ＯＬ　（全 60 頁）

(21)出願番号　特願平9－200570

(22)出願日　平成９年(1997)７月25日

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号

(72)発明者　伊達　厚
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内

(72)発明者　藤原　隆史
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内

(72)発明者　前田　忠昭
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内

(74)代理人　弁理士　大塚　康徳　（外２名）

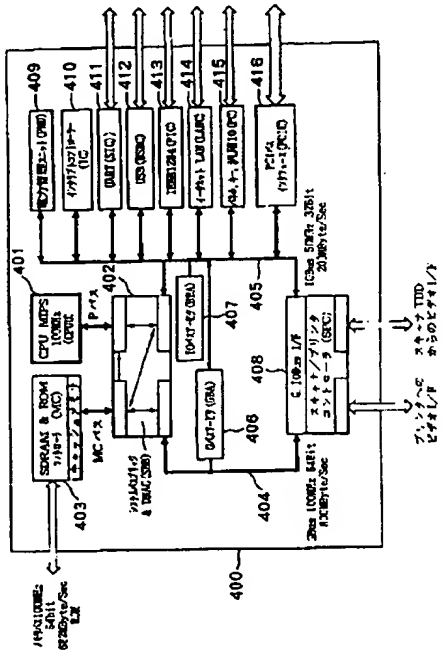
最終頁に続く

(54)【発明の名称】　バス管理装置及びそれを有する複合機器の制御装置

(57)【要約】

【課題】画像データ等、大量のデータを効率良く処理する。

【解決手段】システムバスブリッジ（ＳＳＢ）４０２には、ＣＰＵ４０１からのＰバス、システムメモリからのＭＣバス、入出力機器が接続されたＩＯバス、スキャナプリンタコントローラ４０８の画像データを転送するＧバスが接続され、ＳＳＢ４０２は、マスタとしてＰバス、Ｇバス、ＩＯバスのいずれか、スレーブとしてＭＣバス、ＩＯバスのいずれかをマスタの要求に応じて接続する。この際　ＰバスとＩＯバス、ＧバスとＭＣバスは、並列に接続することができる。そのため、ＣＰＵによる入出力機器の使用と並行して、スキャナプリンタコントローラによるメモリのアクセスを行うことができる。



【特許請求の範囲】
【請求項1】　少なくとも1つのバスと、
該バスに接続された複数のバスマスタと、
前記複数のバスマスタそれぞれに対してバス使用権の付与の開始条件と終了条件とを記憶する手段と、
前記複数のバスマスタからバスの使用要求があると、前記条件にしたがって前記複数のバスマスタのバス使用権の付与を開始し、終了するバス調停手段とを備えることを特徴とするバス管理装置。
【請求項2】　前記複数のバスマスタの順序を記憶する第2の記憶手段を更に備え、前記バス調停手段は、前記順序と前記開始条件とを満足したバスマスタにバス使用権の付与を開始し、バス使用権を与えられており、前記終了条件を満足したバスマスタのバス使用権の付与を終了することを特徴とする請求項1に記載のバス管理装置。
【請求項3】　優先順次モードと完全順次モードとを表すモード設定手段を更に備え、前記バス調停手段は、優先順次モードにおいては、前記記憶手段にバス使用権の付与の開始条件と終了条件とが記憶されたバスマスタを、それ以外のバスマスタに対して優先的にバスの使用権を付与し、完全順次モードにおいては、前記記憶手段にバス使用権の付与の開始条件と終了条件とが記憶されていないバスマスタに対してはバスの使用権を付与しないことを特徴とする請求項1または2に記載のバス管理装置。
【請求項4】　少なくとも4つのバスと、
前記バスに接続されたバスマスタと、
前記バス同士の接続を、前記バスのそれぞれに接続されたバスマスタのバスの要求に応じて切り替える切り替え手段とを備えることを特徴とするバス管理装置。
【請求項5】　前記少なくとも4つのバスの内のひとつにはメモリが、他のバスにはそれぞれ少なくとも1つのバスマスタが接続されていることを特徴とする請求項4に記載のバス管理装置。
【請求項6】　前記バスは、メモリに接続されたバスと、プロセッサに接続されたバスと、入出力機器に接続された少なくとも2つのバスを含み、前記切り替え手段は、プロセッサに接続されたバスと入出力機器に接続されたバスとからの接続要求を調停することを特徴とする請求項5に記載のバス管理装置。
【請求項7】　前記切り替え手段は、前記メモリに接続されたバスと前記入出力機器に接続された一方のバスと、前記プロセッサに接続されたバスと前記入出力機器に接続された他方のバスとのそれぞれを、並列的に接続することを特徴とする請求項6に記載のバス管理装置。
【請求項8】　前記切り替え手段は、前記プロセッサからのメモリあるいは入出力に対する接続要求と、前記入出力バスそれぞれからメモリに対する接続要求とを調停し、バス同士の接続を切り替えることを特徴とする請求項6に記載のバス管理装置。
【請求項9】　画像データの入力及び出力を行う画像入出力機器を制御する制御部を更に備え、該制御部は、前記入出力機器に接続された2つのバスに接続され、選択的にいずれかを用いることを特徴とする請求項4に記載のバス管理装置。
【請求項10】　それぞれにバスマスタを備えた少なくとも2つのバスと、
前記バスを介してアクセスされるメモリと、
前記バスそれぞれに接続され、当該バスのバスマスタによるバス要求を調停し、そのいずれかにバス使用権を付与する調停手段と、
前記バスそれぞれについてバス使用権を付与された複数のバスマスタが同一のあて先に書き込みを行う場合、それらバスマスタのうち、もっとも早くバス要求を出したバスマスタ以外のバスマスタに対するバス使用権を停止させるよう、前記調停手段に対して通知するバス同期手段とを備えることを特徴とするバス管理装置。
【請求項11】　前記バス同期手段は、前記各バスのバスマスタがバス要求を出した時刻を記憶し、バス要求とともに指定されたアドレスを比較して一致した場合には前記記憶された時刻を比較し、もっとも速い時刻のバス要求でないバス要求を出したバスマスタを停止させることを特徴とする請求項10に記載のバス管理装置。
【請求項12】　それぞれがバスの調停手段を有する少なくとも2つのバスと、
前記バスに接続されたバスマスタと、
前記バスの状態と、前記バスマスタから出されるバス要求の情報とを判定し、前記バスのうちいずれを使用するか決定する決定手段とを備えることを特徴とするバス管理装置。
【請求項13】　前記バス要求の情報には、バス要求の優先度とデータの転送先と転送サイズとを含み、前記バスの状態には、各バスの使用状態を含み、前記決定手段は、要求された前記転送先と転送サイズとが許されるバスがひとつの場合にはそのバスを使用するものと決定し、複数ある場合には、転送速度の速い方のバスを優先的に使用するよう決定することを特徴とする請求項12に記載のバス管理装置。
【請求項14】　単一の半導体基板上に形成されてなることを特徴とする請求項1乃至13のいずれかに記載のバス管理装置。
【請求項15】　連続したロケーションへのデータ転送を行うバーストモードを支援するメモリと、
前記メモリの前段にあって、該メモリと交換されるデータを一旦蓄積するキャッシュメモリを備えたメモリ制御手段とを備え、
前記メモリ制御手段は、前記メモリへのデータ転送がバーストモードであればキャッシュを介さずに直接前記メモリへデータを転送し、シングルモードであれば一旦キ

キャッシュにデータを書き込むようキャッシュメモリを制御することを特徴とするメモリ管理装置。

【請求項16】　連続したロケーションへのデータ転送を行うバーストモードを支援するメモリと、

前記メモリの前段にあって、該メモリと交換されるデータを一旦蓄積するキャッシュメモリを備えたメモリ制御手段と、

前記メモリへアクセスする複数のバスマスタとを備え、

前記メモリ制御手段は、前記メモリへのデータ転送をしようとするバスマスタに応じて、キャッシュを介さずに直接前記メモリへデータを転送するか、一旦キャッシュにデータを書き込んでメモリへデータを転送するか制御することを特徴とするメモリ管理装置。

【請求項17】　前記メモリ制御手段は、前記バスマスタの識別子と、それに対応して、キャッシュを介さずに直接前記メモリへデータを転送するか、一旦キャッシュにデータを書き込んでメモリへデータを転送するかの別を記憶する再書込み可能なテーブルを備えることを特徴とする請求項16に記載のメモリ管理装置。

【請求項18】　前記メモリは、シンクロナスDRAMであることを特徴とする請求項15乃至17のいずれかに記載のメモリ管理装置。

【請求項19】　前記メモリ制御手段は、単一の半導体基板上に形成された電気回路であることを特徴とする請求項15乃至18のいずれかに記載のメモリ管理装置。

【請求項20】　制御部により制御された複数の回路ブロックを含む電気回路の消費電力を制御する電力管理装置であって、

各ブロックの動作状態を監視する状態監視手段と、

動作状態にあるブロックの消費電力を加算する加算手段と、

加算された電力を所定の閾値と比較し、閾値を越えたことを前記制御部に通知する通報手段とを備えることを特徴とする電力管理装置。

【請求項21】　前記通報手段は、前記閾値として互いに異なる複数の値を有し、各閾値ごとの比較結果に応じて、通知を行うことを特徴とする請求項20に記載の電力管理装置。

【請求項22】　経過時間を測定する計時手段を更に備え、前記通報手段は、前記通知を行ってから、各ブロックの動作状態が変わらないまま所定時間経過すると再度通知を繰り返すことを特徴とする請求項20または21に記載の電力管理装置。

【請求項23】　前記閾値は、前記制御部により書き換えられることを特徴とする請求項20に記載の電力管理装置。

【請求項24】　請求項1乃至14のいずれかに記載のバス管理装置を備え、該バス管理装置により、画像入力装置及び画像出力装置が接続されたバスを管理することを特徴とする複合機器の制御装置。

【請求項25】　請求項15乃至19のいずれかに記載のメモリ管理装置を備えることを特徴とする複合機器の制御装置。

【請求項26】　請求項20乃至23のいずれかに記載の電力管理装置を備えることを特徴とする複合機器の制御装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、スキャナ等の画像入力装置と、プリンタ等の画像出力装置とを効率的に制御する複合機器の制御装置に関する。

【0002】

【従来の技術】従来、スキャナなどの画像入力装置と、プリンタなどの画像出力装置を組み合わせた、複写機やファクシミリ、あるいはそれらを単体として備えたコンピュータシステムなどが実用化されている。こういったシステムでは、画像データを扱うために、膨大な量のデータを効率的に処理する必要がある。

【0003】

【発明が解決しようとする課題】こういったシステムにおいては、データ転送のために複数のバスマスタによるDMA転送が用いられる。複数のバスマスタが順次に処理を行うことが必要な場合、例えば、メモリ上のデータに対して、処理A（バスマスタ1）を行った後、処理B（バスマスタ2）を行い、その処理後のデータをバスマスタ4に送るといった一連の処理を考える。

【0004】通常このような処理を行う場合、それぞれバスマスタが、メモリからデータをリードし、処理後のデータをメモリに書き戻すというDMA機能があれば、ソフトウェアはまず、バスマスタ1に対して処理Aを行うようにDMAをセットする。バスマスタ1はすべての処理を終了後、プロセッサに対して割り込みを上げ、処理の終了を行うようにDMAをセットし、処理の終了後、今度はバスマスタ4に対してメモリからデータをリードするようにセットを行う。このように一連の処理を行うためには、ソフトウェアによりひとつの処理が終了したのを確認してから次の処理をスタートさせるというように処理を行っていく必要があった。

【0005】このようにこれまではそれぞれの処理を行う度にソフトウェアが介入する必要があり、また処理毎にいちいちメモリに書き戻してやる必要があり、無駄な処理が多いという第1の問題があった。

【0006】また、大量のデータを扱うために、単一のバスではバスの転送能力がボトルネックとなっていた。これを解消するために、バスを2重化して転送能力の増強を図ったシステムもあった。

【0007】しかしながら、複数のバスを有するシステムであっても、バスの構成が柔軟でなく、大量のデータを転送する場合などには、十分な転送能力が引き出されてはいなかったという第2の問題点があった。

【０００８】また、従来のバスは単一であるのが普通であり、複数のバスマスタが同一のメモリアドレスに書き込みを行おうとした場合、バス使用権を獲得した順にメモリへデータを書き込むことが保証できた。

【０００９】しかしながら、複数のバスが存在し、これらのバスのバス調停（アービトレーション）と、複数のバスのどれか1つのバスをメモリへ接続することを独立に制御するシステム構成では、複数のバスの接続された複数のバスマスタが同時に同一のメモリ空間へ書き込みを行う可能性があり、書き込みの順序は、バスの調停によりバスの使用権を得た順序とはならない可能性があるという第3の問題点があった。

【００１０】更に、データを効率的に処理するために、従来キャッシュメモリの使用が行われている。

【００１１】しかしながら従来のキャッシュ制御は、転送先メモリのアドレス情報を基にキャッシュのオン／オフが行われていた。このため、大量のデータをキャッシュ許可のメモリ空間に転送すると、この大量のデータがキャッシュされ、すべて新たなデータで書き換えられてしまい、他のデバイスがメモリにアクセスする場合にはキャッシュミスとなる可能性が高い。これはキャッシュの容量を増やせば解決するかに見えるが、それは大幅な製造コストの上昇につながる。また、特に、プリント時などには、読み出された大量の画像データはそのままプリンタエンジンに渡されてそのデータがキャッシュされたとしても2度と使用されることはない。このようにむやみにキャッシュすることは、逆にキャッシュヒット率を低下させることになる。このように、キャッシュを効率良く利用することができない場合があるという第4の問題点があった。

【００１２】さらに、複数のバスを用いたシステムでは、複数のバスを使用できるバスマスタは、どのバスを使用するか決定する必要がある。従来はアクセスする相手先が決まればそれに応じてバスも決定していた。

【００１３】しかしながら、使用するバスが相手先に応じて固定されているため、各バスの転送速度や使用率等を考慮した効率的なバスの使用が行えないという第5の問題点があった。

【００１４】さらに、このようなシステムをひとつの半導体チップに集積すると大量の熱が発生し、パッケージが熱で破壊されるおそれがあるという第6の問題点があった。

【００１５】本発明は上記第1の問題に鑑みてなされたもので、処理毎のソフトウェアの介入が必要なく全体としての処理速度を向上させるバス管理装置及びそれを有する複合機器の制御装置を提供することを第1の目的とする。

【００１６】また、本発明は上記第2の問題に鑑みてなされたもので、バスの構成を柔軟にし、最適なバスを選んでデータの転送を行えるバス管理装置及びそれを有する複合機器の制御装置を提供することを第2の目的とする。

【００１７】また、本発明は上記第3の問題に鑑みてなされたもので、複数のバスのそれぞれに接続されたバスマスタからのバスの使用権を得た順にメモリにアクセスできるバス管理装置及びそれを有する複合機器の制御装置を提供することを第3の目的とする。

【００１８】また、本発明は上記第4の問題に鑑みてなされたもので、キャッシュの使用効率を向上させるメモリ管理装置及びそれを有する複合機器の制御装置を提供することを第4の目的とする。

【００１９】また、本発明は上記第5の問題に鑑みてなされたもので、各バスマスタが使用するバスを動的に決定してバスの効率を向上させたバス管理装置及びそれを有する複合機器の制御装置を提供することを第5の目的とする。

【００２０】また、本発明は上記第6の問題に鑑みてなされたもので、回路の動作状態を監視して消費電力を抑制し、大量の熱を発生を抑制する電力管理装置及びそれを有する複合機器の制御装置を提供することを第6の目的とする。

【００２１】

【課題を解決するための手段】上記の問題を解決するために、本発明のバス管理装置は次のような構成からなる。すなわち、少なくとも1つのバスと、該バスに接続された複数のバスマスタと、前記複数のバスマスタそれぞれに対してバス使用権の付与の開始条件と終了条件とを記憶する手段と、前記複数のバスマスタからバスの使用要求があると、前記条件にしたがって前記複数のバスマスタのバス使用権を付与を開始し、終了するバス調停手段とを備える。

【００２２】あるいは、少なくとも4つのバスと、前記バスに接続されたバスマスタと、前記バス同士の接続を、前記バスのそれぞれに接続されたバスマスタのバスの要求に応じて切り替える切り替え手段とを備える。

【００２３】あるいは、それぞれにバスマスタを備えた少なくとも2つのバスと、前記バスを介してアクセスされるメモリと、前記バスそれぞれに接続され、当該バスのバスマスタによるバス要求を調停し、そのいずれかにバス使用権を付与する調停手段と、前記バスそれぞれについてバス使用権を付与された複数のバスマスタが同一のあて先に書き込みを行う場合、それらバスマスタのうち、もっとも早くバス要求を出したバスマスタ以外のバスマスタに対するバス使用権を停止させるよう、前記調停手段に対して通知するバス同期手段とを備える。

【００２４】あるいは、それぞれがバスの調停手段を有する少なくとも2つのバスと、前記バスに接続されたバスマスタと、前記バスの状態と、前記バスマスタから出されるバス要求の情報とを判定し、前記バスのうちいずれを使用するか決定する決定手段とを備える。

【0025】あるいは、本発明のメモリ管理装置は次のような構成からなる。すなわち、連続したロケーションへのデータ転送を行うバーストモードを支援するメモリと、前記メモリの前段にあって、該メモリと交換されるデータを一旦蓄積するキャッシュメモリを備えたメモリ制御手段とを備え、前記メモリ制御手段は、前記メモリへのデータ転送がバーストモードであればキャッシュを介さずに直接前記メモリへデータを転送し、シングルモードであれば一旦キャッシュにデータを書き込むようキャッシュメモリを制御する。

【0026】あるいは、連続したロケーションへのデータ転送を行うバーストモードを支援するメモリと、前記メモリの前段にあって、該メモリと交換されるデータを一旦蓄積するキャッシュメモリを備えたメモリ制御手段と、前記メモリへアクセスする複数のバスマスタとを備え、前記メモリ制御手段は、前記メモリへのデータ転送をしようとするバスマスタに応じて、キャッシュを介さずに直接前記メモリへデータを転送するか、一旦キャッシュにデータを書き込んでメモリへデータを転送するか制御する。

【0027】あるいは、本発明の電力管理装置は次のような構成からなる。すなわち、制御部により制御された複数の回路ブロックを含む電気回路の消費電力を制御する電力管理装置であって、各ブロックの動作状態を監視する状態監視手段と、動作状態にあるブロックの消費電力を加算する加算手段と、加算された電力を所定の閾値と比較し、閾値を越えたことを前記制御部に通知する通報手段とを備える。

【0028】あるいは、本発明の複合機器の制御装置は次のような構成からなる。すなわち、上記いずれかに記載のバス管理装置を備え、該バス管理装置により、画像入力装置及び画像出力装置が接続されたバスを管理する。

【0029】あるいは、上記いずれかに記載のメモリ管理装置を備える。

【0030】あるいは、上記いずれかに記載の電力管理装置を備える

【0031】

【発明の実施の形態】次に本発明の実施の形態として、プロセッサコア、プロセッサ周辺コントローラ、メモリコントローラ、スキャナー/プリンタコントローラ、PCIインターフェースなどを内蔵したシングルチップ・スキャニング・プリンティングエンジンである"DoEngine"を説明する。

【0032】1．DoEngine概要

DoEngineは、MIPSテクノロジー社のR4000プロセッサとコンパチブルなプロセッサコア、プロセッサ周辺コントローラ、メモリコントローラ、スキャナー/プリンタコントローラ、PCIインターフェースなどを内蔵したシングルチップ・スキャニング・プリンティングエンジンである。高速並列動作、ビルディングブロック手法を採用し実装される。

【0033】プロセッサシェル（プロセッサコアを含むプロセッサ周辺回路の総称）内には最大でインストラクション、データそれぞれ16Kバイトの計32Kバイトのキャッシュメモリ、FPU（浮動小数点演算ユニット）、MMU（メモリ管理ユニット）、ユーザー定義可能なコプロセッサなどを内蔵することが可能である。

【0034】PCIバスインターフェースを有するので、PCIバススロットを有するコンピュータシステムと共に用いることができる。また、PCIサテライト構成に加え、PCIホストバスブリッジ構成にてPCIバスコンフィギュレーションを発行することが可能であり、安価なPCI周辺デバイスと組み合わせることにより、マルチファンクションペリフェラル（複合機能周辺機器）のメインエンジンとして使用することも可能である。さらにPCIバスインターフェースを有するレンダリングエンジン、圧縮・伸長エンジンと組み合わせることも可能である。

【0035】チップ内部に汎用IOコアを接続するIOバス、及び、画像データ転送に最適化したグラフィックバス（Gバス:Graphics Bus)の2系統の独立したバスを有し、メモリ、プロセッサとこれらのバスをクロスバスイッチを介して接続することにより、マルチファンクションシステムにおける同時動作に必須の、並列性の高い高速データ転送を実現している。

【0036】メモリには、画像データに代表される、連続したデータ列のアクセスに対し、最高のコストパフォーマンスを有するシンクロナスDRAM（SDRAM）をサポートし、SDRAMのバーストアクセス高速データ転送のメリットを享受できない小さなデータ単位でのランダムアクセスにおける性能低下を最小に抑えるために、8Kバイトの2ウェイセットアソシアティブ・メモリフロントキャッシュをメモリコントローラ内に備える。メモリフロントキャッシュは、すべてのメモリライトに対するバススヌープが難しい、クロスバスイッチを採用したシステム構成においても、複雑な機構なしに、キャッシュメモリによる高性能化が達成出来る方式である。また、リアルタイムデータ転送（機器制御）が可能な、プリンタ及びスキャナーとのデータインターフェース(Video Interface)を有し、さらにハードウェアによる、機器間同期のサポート及び画像処理を行う事により、スキャナ、プリンタ分離型の構成においても、高品質で高速なコピー動作の実現が可能な構成となっている。

【0037】なお、DoEngineは、コアが3．3Vで動作し、IOは5Vトレラントである。

【0038】図1及び図2，図3は、DoEngineを用いた装置あるいはシステムの構成例を示している。図1は分離構成型であり、パーソナルコンピュータ10

2には、それが備えるPCIインターフェースを介してDoEngineを備えたローカルボード101が装着される。ローカルボード101にはDoEngineのほか、後述するメモリバスを介してDoEngineと接続されたメモリと、色処理回路（チップ）が設けられている。このローカルボード101を介して、高速スキャナ103とカラー／モノクロプリンタ104とがパーソナルコンピュータ102に接続される。この構成により、パーソナルコンピュータの制御のもとで、ローカルボード101により、スキャナ103から入力された画像情報を処理し、プリンタ104から出力させることができる。

【0039】また、図2及び図3はスキャナ203とプリンタ202とを一体に組み込んだ例で、図2は通常の複写機に類する構成を、図3（a）はファクシミリ装置などの構成を示している。図3（b）は、図3（a）をコントロールするコンピュータを示している。

【0040】これらのうち、図1，2は、PCIインターフェースを介して接続された外部のCPUによりDoEngineが制御されるというスレーブモードで使用されている例であり、図3は、DoEngineのCPUが主体となり、PCIインターフェースを介して接続されたデバイスを制御するマスタモードで使用されている例である。

【0041】表1として、DoEngineの仕様を示す。外部インターフェースとして、PCI，メモリバス，ビデオ，汎用入出力，IEEE1284，RS232C，100baseT／10baseT，LCDパネル及びキーを備えるが、更にUSBを有していてもよい。内蔵ブロックとして、CPUコアに加えて、1次キャッシュ，キャッシュ付きメモリコントローラ，コピーエンジン，IOバスアービタ，グラフィックバスアービタなどを備えている。また、DMAコントローラはチャネル数が5であり、グラフィックバス，IOバスともにアービトレーションは、優先度付きの先着順処理方式で行われる。

【0042】

【表1】

項目	概要	スペック
チップ	動作周波数	内部100MHz、内部バス及びメモリバス100MHz
	パッケージ	313ピンBGA
	外部インターフェース	PCI
		メモリバス
		ビデオ
		汎用入出力
		IEEE1284
		RS232C
		(USB)
		LAN 100/10 baseT
		LCDパネル&キー
	内蔵ブロック	CPUコア
		1次キャッシュ
		MMU
		ICU
		システムバスブリッジ
		キャッシュ付メモリコントローラ
		COPYエンジン
		PLL
		電源制御ユニット
		IOバスアービタ
		グラフィックバスアービタ
DMAコントローラ	チャネル数	5チャネル
	最大転送速度（ピーク）	200M byte/Sec@50MHz
	転送可能経路	内部出力ブロック↔ローカルメモリ
メモリ及びバス制御	サポートメモリ	SDRAM
	データ幅	64ビット
	最大メモリ容量	1Gbyte
	最大メモリバス転送速度	682M byte/Sec
グラフィックバス	アービトレーション方式	優先度付先着順処理
	最大バス転送速度	800M byte/Sec
	バス幅	64Bit,100MHz
PCIバス	PCIバス形式	Rev2.1,32Bit,33M PCI
	マスター時転送速度	Read 96MByte/Sec,Write 88MByte/Sec
	スレーブ時転送速度	Read 101MByte/Sec,Write 111MByte/Sec
IOバス	アービトレーション方式	優先度付先着順処理
	最大バス転送速度	200M byte/Sec
	バス幅	32Bit,50MHz

【0043】2．DoEngineの構成及び動作
本章では、DoEngine の総論に加え、各機能ブロック毎のブロック図、概要、詳細、コアインターフェース、タイミング図などを解説する。
【0044】2．1．DoEngineのチップ構成
図4として、DoEngineのブロック図を示す。DoEngine400は次世代複合機能周辺機器（システム）(MFP:Multi Function Peripheral or MFS:Multi Function System)の主たるコントローラとして設計、開発された。
【0045】CPU（プロセッサコア）401として、MIPSテクノロジー社のMIPSR4000コアを採用する。プロセッサコア401内には、8Kバイトずつのインストラクション、データのキャッシュメモリ、MMUなどが実装される。プロセッサコア401は、64ビットのプロセッサバス（Pバス）を介して、システム

・バス・ブリッジ（ＳＢＢ）４０２に接続される。ＳＢＢ４０２は４×４の６４ビットクロスバスイッチであり、プロセッサコア４０１の他に、キャッシュメモリを備えたＳＤＲＡＭやＲＯＭを制御するメモリコントローラ４０３と専用のローカルバス（ＭＣバス）で接続されており、さらに、グラフィックバスであるＧバス４０４、ＩＯバスであるＩＯバス４０５と接続され、全部で３つのバスに接続される。システムバスブリッジ４０２は、これら４モジュール間を、可能な限り、同時平行接続を確保することができるように設計されている。

【００４６】Ｇバス４０４はＧバスアービタ（ＧＢＡ）４０６により協調制御されており、スキャナやプリンタと接続するためのスキャナ／プリンタコントローラ（ＳＰＣ）４０８に接続される。また、ＩＯバス４０５は、ＩＯバスアービタ（ＢＢＡ）４０７により協調制御されており、ＳＰＣ４０８のほか、電力管理ユニット（ＰＭＵ）４０９、インタラプトコントローラ（ＩＣ）４１０、ＵＡＲＴを用いたシリアルインターフェースコントローラ（ＳＩＣ）４１１、ＵＳＢコントローラ４１２、ＩＥＥＥ１２８４を用いたパラレルインターフェースコントローラ（ＰＩＣ）４１３、イーサネットを用いたＬＡＮコントローラ（ＬＡＮＣ）４１４、ＬＣＤパネル、キー、汎用入出力コントローラ（ＰＣ）４１５、ＰＣＩバスインターフェース（ＰＣＩＣ）４１６にも接続されている。

【００４７】２．２．プロセッサシェル

プロセッサシェルとは、プロセッサコアに加えＭＭＵ（Memory Management Unit）、命令キャッシュ、データキャッシュ、ライトバックバッファ及び掛け算ユニットを含んだブロックを指す。

＜キャッシュメモリ＞図５に示したように、キャッシュメモリコントローラは、無効（Invalid）、有効かつクリーン（Valid Clean：キャッシュが更新されていない）、有効かつダーティ（Valid Dirty：キャッシュが更新されている）の３つのステートキャッシュを管理する。この状態に応じて、キャッシュは制御される。

【００４８】２．３．インタラプトコントローラ

図６にインタラプトコントローラ４１０のブロック図を示す。

【００４９】インタラプトコントローラ４１０は、ＩＯバスインターフェース６０５を介してＩＯバス４０５に接続され、ＤｏＥｎｇｉｎｅチップ内の各機能ブロック及び、チップ外部からのインタラプトを集積し、ＣＰＵコア４０１がサポートする、６レベルの外部インタラプト及び、ノンマスカブルインタラプト（ＮＭＩ）に再分配する。各機能ブロックとは、電力管理ユニット４０９、シリアルインターフェースコントローラ４１１、ＵＳＢコントローラ４１２、パラレルインターフェースコントローラ４１３、イーサネットコントローラ４１４、汎用ＩＯコントローラ４１５、ＰＣＩインターフェースコントローラ４１６、スキャナ／プリンタコントローラ１０８などである。

【００５０】この際、ソフトウェアコンフィギュレーション可能なマスクレジスタ（Int Mask Logic 0－5）６０２により、各要因毎に割り込みをマスクをすることが可能である。また、外部インタラプト入力は、選択的エッジ検出回路６０１により、信号線ごとに、エッジセンス／レベルセンスを選択することが出来る。要因レジスタ（Detect and set Cause Reg 0－5）６０３は、各レベルごとに、どのインタラプトがアサートされているかを示すとともに、ライト動作を行うことで、レベルごとにクリアを行う事が出来る。

【００５１】各レベルの割込み信号は、各レベル毎に少なくともひとつの割り込みがあれば割込み信号を出力すべく、論理和回路６０４により論理和として出力される。なお、各レベル内での複数要因間のレベル付けはソフトウェアにて行う。

【００５２】２．４．メモリコントローラ

図７は、メモリコントローラ４０３のブロック図である。メモリコントローラ４０３は、メモリコントローラ専用のローカルバスであるＭＣバスにＭＣバスインターフェース７０１を介して接続され、最大１ギガバイトのシンクロナスＤＲＡＭ（ＳＤＲＡＭ）と、３２メガバイトのフラッシュＲＯＭあるいはＲＯＭをサポートする。ＳＤＲＡＭの特徴であるバースト転送時の高速性を活かすため、６４（１６×４）バースト転送を実現する。また、ＣＰＵやＩＯバスよりの連続したアドレスのシングル転送を考慮し、メモリコントローラ内にＳＲＡＭ（メモリフロントキャッシュ）７０２を内蔵し、ＳＤＲＡＭへ直接シングル転送を行うことを可能な限り回避して転送効率を向上させる。メモリコントローラ－ＳＤＲＡＭ間のデータバス幅は、信号ramData及びramParをあわせて７２ビット（このうち８ビットの信号ramParはパリティ）、フラッシュＲＯＭ間のデータバスfntromData，prgromDataの幅は３２ビットとする。

【００５３】２．４．２．構成及び動作

メモリコントローラの各部はこれから説明するような構成となっている。

＜ＭＣバスインターフェース（７０１）＞ＭＣバスは、ＳＢＢ４０２－メモリコントローラ４０３間の専用のバスであり、またＳＢＢ内部の基本バスとして用いられている。

【００５４】ＣＰＵ４０１とバスブリッジ４０２とを接続する専用バスＰＢｕｓのバースト転送が４バーストのみ規定しているのに対し、ＭＣバスにおいては１６バースト×４までの転送を追加している。このためにバースト長を示す信号としてｍＴＴｙｐｅ［６：０］を新たに定義した。

（ＭＣバス信号の定義）

ＭＣバスの各信号は下記の通り定義される。

・mClk（出力）　　…　MCバスクロック
・mAddr[31:0]（出力）　…　MCバスアドレス
32ビットのアドレスバスであり、mTs_L がアサートされた時点からmBRdy_Lがアサートされるまで保持される。
・mDataOut[63:0]（出力）…　MCバスデータ出力
64ビットの出力データバスであり、mDataOe_Lがアサートされている時のみ有効である。
・mDataOe_L（出力）　…　MCバスデータ出力イネーブル
mDataOut[63:0]が有効であることを示す。またその転送がWriteであることを示す。
・mDataIn[63:0]（入力）…　MCバスデータ入力
64ビットの入力データバスであり、mBRdy_LがアサートされているmClkの立ち上がりでサンプリングされる。
・mTs_L（出力）　…　MCバストランザクション開始ストローブ
転送が開始したことを示す。転送の最初の1クロックの間だけアサートされる。転送が1クロックで終了し、次の転送がすぐに始められるのならばmTs_Lは引続きアサートされたままになる。
・mTType[6:0]（出力）…　MCバストランザクションタイプ
MCバス上の転送のタイプを示す。シングル転送時はその転送の間、バースト転送時は最初の転送(beat)の間保持される。上位3ビットがソース（マスタ）をあらわし、下位ビットがシングル／バースト長をあらわす。タイプには次のようなものがある。

mTType[6:4]	信号源
001	CPU
010	IOバス
100	Gバス

mTType[3:0]	シングル／バースト長
1xxx	シングル（1　8byte）
0001	2バースト
0010	4バースト
0011	6バースト
0100	8バースト
0101	16バースト
0110	2×16バースト
0111	3×16バースト
0000	4×16バースト

・mBE_L[7:0]（出力）　…　MCバストランザクションバイトイネーブルシングル転送時、64ビットデータバス上の有効なバイトレーンを示す。バースト転送時はWrite時のみ有効であり、Read時は無視される。
・mBRdy_L（入力）　…MCバスレディ
現在の転送(beat)が終了したことを示す。
・mTPW_L（出力）　…　次トランザクションがIn-page write（ページ内書き込み）
次の転送が同じページ（同じRowアドレス）のWriteであることを示し、最大4個までのWriteを続けることができる。ページサイズはあらかじめコンフィギュレーションレジスタに設定しておく。
・mBPWA_L（入力）　…　バスのページ内書き込み許可
MCバススレーブ（メモリコントローラ）がページ内書き込みトランザクションを許可するかどうかを示し、mBRdy_Lと同じクロックでサンプリングされる。この時mBPWA_LがディアサートされていればmTPW_Lは無意味となる。
・mBRty_L（入力）　…　バスリトライ
MCバススレーブ（メモリコントローラ）がアクセスを未実行のまま終了させる場合にアサートし、少なくとも1サイクル以上のアイドルの後に再試行しなければならないことを示す。（もしmBRdy_LとmBRty_Lが同時にアサートされた場合は、mBRty_Lが優先される。）
・mBerr_L（入力）　…　バスエラー
パリティエラーやその他のバスエラーが発生した場合にアサートされる。
【００５５】なお、上述した入出力の別はＳＢＢからみての定義である。
（ＭＣバストランザクション）ＭＣバストのトランザクションとしては、以下のトランザクションをサポートする。
①ベーシックトランザクション(1,2,3,4,8バイト Read/Write)
mBE_L[7:0]信号に従い、１，２，３，４，８バイトのシングルトランザクションをサポートする。
②バーストトランザクション
（CPUからの）4-ダブルワードバーストまでのトランザクションをサポートする。
③Gバスからの16-ダブルワードバースト×4までのトランザクションをサポートする。
④In-page write（ページ内書き込み）トランザクション
mTPW_Lで示される同一ページ内の書き込みに関して、連続的なWriteアクセスをサポートする。
⑤バスリトライ
メモリコントローラ内の制限によりメモリアクセスができない場合は、mBRty_L信号をアサートし、バスリトライを通知する。
＜ＳＤＲＡＭコントローラ（７０５）＞メモリコントローラ４０３は、次のような構成を有するＳＤＲＡＭを以下のように制御する。
【００５６】（ＤＲＡＭ構成）ＤＲＡＭの構成としては、x4,x8,x16ビットタイプの１６／６４メガビットＳ

DRAMを64ビットデータバスで8バンク制御することができる。

【0057】

【表2】

バンク内のデバイス数	デバイス構成	Rowビット(バンク選択ビットを含む)×Columnビット	バンクサイズ	最大メモリ(8バンク)
16(64Mbit Type)	16M×4	14×10	128 Mbyte	1 Gbyte
8(64Mbit Type)	8M×8,9	14×9	64 Mbyte	512 Mbyte
4(64Mbit Type)	4M×16,18	14×8	32 Mbyte	256 Mbyte
16(16Mbit Type)	4M×4	12×10	32 Mbyte	256 Mbyte
8(16Mbit Type)	2M×8,9	12×9	16 Mbyte	128 Mbyte
4(16Mbit Type)	1M×16,18	12×8	8 Mbyte	64 Mbyte

【0058】(DRAMアドレスビット構成)DRAMのアドレスビットの割付けについては、64ビットSDRAMの場合にはMA(13:0)を、16ビットSDRAMの場合にはMA(11:0)を使用する。

【表3】

64 Mbit SDRAM

31　30	29---27	26 25	24 ---11	10 --- 3	2 --- 0
0　0	CS	C9 C8	R13---R0	C7 - C0	BS

16 Mbit SDRAM

21　28	27---25	24 23	22 ---11	10 --- 3	2 --- 0
0　0	CS	C9 C8	R11---R0	C7 - C0	BS

0	Zero
CS	Chip Select
C9 - C0	Column Address ×8bit SDRAMの場合、C8は無視 ×16bit SDRAMの場合、C9,C8は無視
R13 - R0	Row Address 16M SDRAMのR11はSDRAM内のバンクセレクトに用いられる。 64M SDRAMで4Bank構成の場合R12,R13が 2Bank構成の場合R13がバンクセレクトに用いられる。
BS	Byte Select

(SDRAMプログラマブル構成(モードレジスタ))

SDRAMは内部にモードレジスタを持ち、モードレジスタ設定コマンドを用いて下記の項目を設定する。

①バースト長

バースト長は、1，2，4，8，フルページのいずれかが設定可能であるが、CPUからのバースト転送長が4であることから、バースト長4が最適である。Gバスからの16バースト以上の転送は、Read/Writeコマンド(オートプリチャージ無し)を連続して発行することにより実現する。

②ラップタイプ(Wrap Type)

バースト転送時のアドレスのインクリメント順を設定する。「シーケンシャル」または「インターリーブ」のどちらかが設定可能である。

③CASレーテンシ

CASレーテンシは、1，2，3のいずれかが設定可能であり、使用するSDRAMのグレードと動作クロックにより決定される。

【0059】(SDRAMコマンド)SDRAMに対して以下のコマンドをサポートする。各コマンドの詳細は、SDRAMデータブックに記載されている。

・モードレジスタ設定コマンド
・アクティブコマンド
・プリチャージコマンド
・ライトコマンド
・リードコマンド
・CBR(Auto)リフレッシュコマンド
・セルフリフレッシュ開始コマンド
・バーストストップコマンド
・NOPコマンド

(SDRAMリフレッシュ)SDRAMは、2048サイクル/32ms(4096/64ms)であるので、16.625nsおきにCBRリフレッシュコマンドを発行する。メモリコントローラは設定可能なリフレッシュカウンタを持ち、自動的にCBRリフレッシュコマンドを発行する。Gバスからの16-バーストx nの転送中は、リフレッシュ要求を受け付けない。したがって、リフレッシュカウンタは16-バーストx 4転送の時間だけ余裕を持った値を設定しなければならない。また、セルフリフレッシュをサポートする。このコマンドを発行すると、パワーダウンモード(rame1ke_L=Low)時にセルフリフレッシュが続行される。

【0060】(SDRAM初期化)メモリコントローラはパワーオンリセット後、SDRAMに対して以下の初期化を行なう。すなわち、電源投入後100μsのポーズ期間をおいて、

①プリチャージコマンドを用いて全バンクをプリチャージする。

②SDRAMのモードレジスタを設定する。

③オートリフレッシュコマンドを用いて、リフレッシュを8回行う。

≪フラッシュROMコントローラ（７０４）≫フラッシュROMコントローラ７０４は、romAddr[23:2]のアドレス信号と４個のチップセレクト(romCs_L[3:0])信号をサポートする。アドレス信号romAddr2～romAddr9はパリティ信号ramPar0～ramPar7とマルチプレクスされ、アドレス信号romAddr10～romAddr23は、ＤＲＡＭアドレスramAddr0～ramAddr13とマルチプレクスされている。

≪ＳＲＡＭコントロール（メモリフロントキャッシュ）≫メインメモリとして用いられるＳＤＲＡＭは、バースト転送は非常に高速であるが、シングル転送においてはその高速性が発揮できない。そこで、メモリコントローラ内にメモリフロントのキャッシュを実装し、シングル転送の高速化をはかる。メモリフロントキャッシュは、キャッシュコントローラ７０６とＳＲＡＭ７０２より構成される。MCバスに定義されたmTType[6:0]信号により、その転送マスターと転送長を知ることができるので、各マスターごと、あるいは転送長ごとにキャッシュのオン／オフが設定可能である。キャッシュの方式は、次の通りである。なお、以下、特に断らない限り、単なるキャッシュあるいはキャッシュメモリという呼称は、プロセッサコアに内蔵されたキャッシュではなく、メモリコントローラが内蔵するメモリフロントキャッシュを指すものとする。

・2ウェイセットアソシアティブ
・8kバイトデータＲＡＭ
・128 x 21 x 2 タグＲＡＭ
・LRU(Least Recently Used)アルゴリズム
・ライトスルー
・No Write Allocate

次にキャッシュコントローラ７０６を中心とする詳細なブロック図を図８に示す。

【００６１】（キャッシュの動作）ここで、MCバスよりメモリリード／ライト転送が要求された場合のキャッシュの動作を図８のブロック図と、図９及び図１０に示すフローチャートを用いて説明する。

【００６２】MCバスよりデータ転送が開始されると、その転送の最初にMCバス上で示されるmTType[6:0]によって、その転送がキャッシュオンでおこなうか、オフでおこなうかの判断がおこなわれる。この説明では、転送がシングル転送であればオン、バースト転送であればオフと判断することにする（ステップＳ９０１）。すなわち、mTType(3)が“１”hであれば、シングル転送をあらわすのでキャッシュオンで、“０”hであればバースト転送をあらわすのでキャッシュオフで転送を行う。

【００６３】シングル転送（キャッシュオン）の場合、アドレスlmaddr[31:0]が与えられると、lmaddr[11:5]がインデックスとしてb1_tag_ram８０１，b2_tag_ram８０２，b1_data_ram７０２－ａ，b2_data_ram７０２－ｂ，lru８０３へ与えられ、それぞれのブロックから、入力されたインデックスに対応するバリッドビット"v"及びb1_tag_addr，バリッドビット"v"及びb2_tag_addr，b1_out_data，b2_out_data，lru_inがそれぞれ出力される（ステップＳ９０２）。

【００６４】b1_tag_ram８０１とb2_tag_ram８０２より出力されたb1_tag_addrとb2_tag_addrがそれぞれb1_comparater８０４，b2_comparater８０５でアドレスlmaddr[31:12]と比較され、その結果、すなわちヒットか否かがb1_hit_miss_L,b2_hit_miss_L信号によりキャッシュコントローラ７０６へ知らされ、判定される（ステップＳ９０３）。

【００６５】ここでヒットであれば、リードかライトかが判定される（ステップＳ９０４）。ヒットであるとは、b1_tag_addrとb2_tag_addrのいずれかにアドレスlmaddr[31:12]と一致するものがある場合である。ヒットかつリードの場合にはつぎのようになる。すなわち、b1がヒットであり、要求された転送がリードであった場合は、すでに読み出されているb1_out_dataとb2_out_dataのうちb1_out_dataを選択し、lmaddr[4:3]で示される8バイトのデータをMCバスへ出力する（ステップＳ９０５）。同時にそのインデックスに対応するlruを“０”(=b1ヒット)に書き換えて、転送が終了する。b2がヒットであり、要求された転送がリードであった場合は、すでに読み出されているb1_out_dataとb2_out_dataのうちb2_out_dataを選択し、lmaddr[4:3]で示される8バイトのデータをMCバスへ出力する（ステップＳ９０５）。同時にそのインデックスに対応するlruを“１”h（=b2ヒット)に書き換えて、転送が終了する。

【００６６】一方、ヒットかつライトの場合にはつぎのようになる。すなわち、b1がヒットであり、要求された転送がライトであった場合は、インデックスで示されるb1_data_ram７０２－ａのlmaddr[4:3]で示される8バイトのデータのうちmBE_L[7:0]で示される有効なバイトレーンのみを書き換え、同時にそのインデックスに対応するlruを“０”h(=b1ヒット)に書き換える。またＳＤＲＡＭも同様に書き換えて転送が終了する（ステップＳ９０６）。b2がヒットであり、要求された転送がライトであった場合は、インデックスで示されるb2_data_ram７０２－ｂのlmaddr[4:3]で示される8バイトのデータのうちmBE_L[7:0]で示される有効なバイトレーンのみを書き換え、同時にそのインデックスに対応するlruを“１”h(=b2ヒット)に書き換える。またＳＤＲＡＭも同様に書き換えて転送が終了する（ステップＳ９０６）。

【００６７】一方、b1,b2ともにミスである場合にも、リードかライトか判定される（ステップＳ１００１）。要求された転送がリードであった場合は、そのlmaddr[31:3]で示される8バイトのデータがＳＤＲＡＭより読み出され（ステップＳ１００３）、MCバスへ出力される（ステップＳ１００４）。同時にそのインデックスに対

応するlruが読み出され、“0”hであった場合はb2_data_ramへSDRAMからのデータを書きlruも“1”hへ書き換える。lruが“1”hであった場合はb1_data_ramへSDRAMからのデータを書き、lruも“0”hへ書き換え終了する（ステップS1005）。b1,b2ともにミスであり、要求された転送がライトであった場合は、SDRAMに書き込むだけで転送が終了する（ステップS1002）。

【0068】ステップS901においてバースト転送（キャッシュオフ）の場合、リード、ライトともSDRAMに対してだけ行われ（ステップS907-S909）、キャッシュデータやタグの書き換え等は行わない。

<ROM・RAMインターフェース（707）>ROM・RAMコントローラ707の構成を図11に示す。ブロック1101～ブロック1104により、SDRAMのデータ信号、アドレス信号、パリティ信号が、フラッシュROMのデータ信号、アドレス信号と多重化される。

【0069】2．4．3．タイミングダイアグラム

上述したメモリコントローラ403によるデータの読み出し・書き込み等の処理のタイミングを、図12～図19を用いて説明する。

【0070】図12は、CPUからのバースト読み出しのタイミングを示す。バースト長は4、CASレイテンシは3である。図9のステップS909における処理に相当する。

【0071】図13は、CPUからのバースト書き込みのタイミングを示す。バースト長は4、CASレイテンシは3である。図9のステップS908における処理に相当する。

【0072】図14は、Gバスデバイスからのバースト読み出しのタイミングを示す。Gバスのバースト長は16、SDRAMのバースト長は4、CASレイテンシは3である。図9のステップS909における処理に相当する

【0073】図15は、Gバスデバイスからのバースト書き込みのタイミングを示す。Gバスのバースト長は16、SDRAMのバースト長は4、CASレイテンシは3である。図9のステップS908における処理に相当する。

【0074】図16は、メモリフロントキャッシュにヒットした場合のシングル読み出しのタイミングを示す。読み出されるデータmDataIn[63:0]としては、キャッシュメモリであるb1_data_ram702-aあるいはb2_data_ram702-bから読み出されたb1/b2_out_dataが出力される。SDRAMのバースト長は4、CASレイテンシは3である 図9のステップS905における処理に相当する。

【0075】図17は、メモリフロントキャッシュにヒットしなかった場合のシングル読み出しのタイミングを示す。読み出されるデータmDataIn[63:0]としては、SDRAMから読み出されたデータramData[63:0]が出力される。また、このデータはb1/b2_in_dataとしてキャッシュメモリであるb1_data_ram702 aあるいはb2_data_ram702-bにも書き込まれる。SDRAMのバースト長は4、CASレイテンシは3である。図10のステップS1004及びS1005における処理に相当する。

【0076】図18は、メモリフロントキャッシュにヒットした場合のシングル書き込みのタイミングを示す。書き込まれるデータmDataOut[63:0]は、キャッシュメモリであるb1_data_ram702-aあるいはb2_data_ram702-bに書き込まれるとともに、SDRAMにも書き込まれる。SDRAMのバースト長は4、CASレイテンシは3である。図9のステップS906における処理に相当する。

【0077】図19は、メモリフロントキャッシュにヒットしなかった場合のシングル書き込みのタイミングを示す。書き込まれるデータmDataOut[63:0]は、キャッシュメモリであるb1_data_ram702-aあるいはb2_data_ram702-bには書き込まれず、SDRAMに対してだけ書き込まれる。SDRAMのバースト長は4、CASレイテンシは3である。図10のステップS1002における処理に相当する。

【0078】なお、ここでは、MCバスよりデータ転送が開始されると、その転送の最初にMCバス上で示されるmTType[6:0]によって、転送がシングル転送であればオン、バースト転送であればオフと判断しているが、バースト転送の場合に、さらにバースト長を判定し、バースト長がキャッシュの1ラインよりも小さい場合にはキャッシュオンとし、そうでない場合にはキャッシュオフとして動作するようにしてもよい。

【0079】また、MCバスに、メモリへのデータ転送を要求したバスマスタの識別子を示す信号を含ませることで、メモリコントローラがその識別子を判定し、識別子に応じてキャッシュオン／キャッシュオフの制御を行うこともできる。この場合には、識別子とキャッシュオン／オフの別とを対応させた書換え可能なテーブルを用意し、それを参照してキャッシュのオンオフを切り替えることもできる。このテーブルは、例えば特定のアドレスを割り当ててCPU401などから書換え可能にすることもできる。

【0080】2．5．システムバスブリッジ（SBB）及びIOバス，Gバス

図20としてシステムバスブリッジ（SBB）402のブロック図を示す。

【0081】SBB402は、IOバス（入出力バス），Gバス（グラフィックバス），Pバス（プロセッサローカルバス）及びMCバス間の相互接続をクロスバス

イッチを用いて提供する、マルチチャネル双方向バスブリッジである。クロスバスイッチにより、2系統の接続を同時に確立することが出来、並列性の高い高速データ転送を実現出来る。

【0082】SBB402は、IOバス405と接続するためのIOバスインターフェース2906と、Gバス404と接続するためのGバスインターフェース2006と、プロセッサコア401と接続するためのCPUインターフェーススレーブポート2002と、メモリコントローラ403と接続するためのメモリインターフェースマスターポートを備えるほか、アドレスバスを接続するアドレススイッチ2003，データバスを接続するデータスイッチ2004を含む。また、プロセッサコアのキャッシュメモリを無効化するキャッシュ無効化ユニット2005を備えている。

【0083】IOバスインターフェース2009には、IOバスデバイスからのDMAライトを高速化するライトバッファと、IOバスデバイスのリードを効率化するリードプリフェッチキューを実装する。これらのキュー内に一時的に存在するデータに関するコヒーレンシ管理はハードウェアにて行う。なお、IOバスに接続されたデバイスをデバイスと呼ぶ。

【0084】プロセッサコアは32ビットバスに対するダイナミックバスサイジングをサポートしているが、SBB402ではこれをサポートしない。将来、バスサイジングをサポートしないプロセッサを用いる場合にもSBBに必要な改造を最小限におさえるためである。

【0085】2．5．1．SBB及び各バスの構成及び動作

＜IOバスインターフェース＞図21は、IOバスインターフェースのブロック図である。

【0086】IOバスインターフェース2009は、IOバスとMCバス間の双方向ブリッジ回路である。IOバスはDoEngineの内部汎用バスである。

【0087】IOバスインターフェース2009内には、マスタコントロールブロック2011，スレーブコントロールブロック2010，データインターフェース2012，DMAC2013，IOバスバッファの5ブロックが含まれる。図21上で、DMAC2013は、機能上3個のシーケンサとレジスタブロックに分割される。それら3つのシーケンサのうち、DMAメモリアクセスシーケンサはIOバススレーブコントロールブロック2010に、DMAregシーケンサはIOバスマスターコントロールブロック2011に内蔵される。レジスタブロックであるDMAレジスタはIOバスデータインターフェース2012部に内蔵される。

【0088】またIOバスインターフェース2009はIOバス側からのメモリへの書き込み時、及びDMAによるデバイスからメモリへの転送が起こった時に、キャッシュ無効化インターフェースを介してCPUシェル内のデータ、命令両キャッシュの無効化の制御を行う。

【0089】なお、CPUライト時のライトバックバッファはIOバスインターフェースには実装しないが、IOバス上の外部マスタライト時のライトバッファを実装する。これにより、バースト転送でない、連続した外部マスタからの書き込みが高速化される。このライトバッファのフラッシュは、IOバスアービタ407によるメモリへの接続が許された時点で行われる。IOバスマスタリードのライトバッファバイパスは行わない。

【0090】また、外部マスタのリードプリフェッチキューを実行する。これにより、外部マスタからの連続した、データストリームの読み出しの高速化を図る。リードバッファの無効化は、

1．IOバスの新たなリードがバッファにヒットしなかった場合、

2．CPUからメモリへのライトが行われた場合。

3．Gバスからメモリへのライトが行われた場合。

4．IOバスからメモリへのライトが行われた場合、

に行われる。

【0091】またIOバスインターフェース2009にはIOバス405上の各デバイスとメモリ間のDMAコントローラ2013が内蔵される。システムバスブリッジ102にDMAコントローラを内蔵することにより、ブリッジ双方へ、同時にアクセス要求が発行出来，効率的なDMA転送が実現出来る。

【0092】IOバスインターフェース2009はプロセッサ401からのアクセス要求に対して、ダイナミックバスサイジングの使用を要求しない。またIOバスマスタからのメモリアクセス要求時に、メモリコントローラ403からのバスサイジングにも対応しない。すなわち、メモリコントローラはバスサイジングを期待すべきではない。

＜IOバス＞

IOバスはDoEngine内の汎用IOバスであり、以下の仕様を持つ。

・アドレス、データ分離型32ビットバス。

・任意のウェイトサイクルを挿入可能、最短はノーウェイト。

・バーストトランザクションのサポート。

・最大転送速度は、クロックが50MHz時に200Mbyte/Sec。

・バスエラーとバスリトライのサポート。

・複数バスマスタのサポート。

【0093】（IOバス信号定義）以下にバス信号の定義を説明する。各信号毎に、「信号名（英語呼称）：入力元>出力先　(, 3States)…信号の説明」の要領で記載されている。なお、3ステートの項は3ステートの信号に限り記載した。

【0094】bAddr[31:2]（IOBus Address Bus）：Master>;Slabe, 3State…　IOBusアドレスバス。

【００９５】bData[31:0] (IOBus Data Bus) : DataDriver>;DataReceiver, 3State … IOBusデータバス。

【００９６】b(Datadrivername)DataOeReq (IOBus Data Output Enable Request) : Datadriver>;DefaultDriverLogic … 後述する双方向IOバスを実現するため、デフォルトドライバーコントロールロジックへの出力信号である。Datadrivernameを持つデバイスが、バス上にデータをドライブするための要求信号である。データの出力を許可されたデバイスには、デフォルトドライバーコントロールロジックよりb(Datadrivername)DataOe_Lが出力される。Datadriverの例：Pci,Sbb,Jpeg,Spu。

【００９７】b(Datadrivername)DataOe_L (IOBus Data Output Enable) : dfaultDriverLogic>;Datadriver … b(Datadrivername)DataOeReqを出力したデバイスに対し、デフォルトドライバーロジックがデータバスへのデータのドライブを許す場合b(Datadrivername)DataOe_L信号をそのデバイスに対して返す。

【００９８】bError_L (IOBus Bus Error) : Slave>;Master 3State … IOバストランザクションがエラーで終了したことを示す。

【００９９】b(Mastername)BGnt_L (IOBus Grant) : Arbiter>;Master … バスアービトレーションにより当マスタがバスの使用権を得たことを示す。Mastername の例：Pci,Sbb,Jpeg,Spu。

【０１００】bInstNotData (IOBus Instruction/Data Output Indicator) : Master>;Slave, 3State … IOバスマスタがインストラクションフェッチをIOバススレーブに対して行う場合にHighにドライブする。データトランザクションの場合はLowにドライブする。

【０１０１】b(Mastername)CntlOeReq (IOBus Master Control Output Enable Request) : Master>;DefaultDriverLogic … IOバスマスタが、bStart_L,bTx_L,bWr_L,vInstNotDataとbAddr[31:2]を3ステートバス上にドライブしたい場合に、IOBus Output Control Logicに対してアサートする。IOBus Output Control Logic は各マスタからの、bMCntlOeReqに基づきドライブを許すマスタに対し、b(Mastername)CntlOe_L信号を返す。

【０１０２】b(Mastername)CntlOe_L (IOBus Master Control Output Enable) : DefaultDriverLogic>;Master … b(Mastername)CntlOeReqを出力したマスタに対し、デフォルトドライバーロジックがドライブを許す場合b(Mastername)CntlOe_L信号をそのマスタに対して返す。

【０１０３】bRdy_L (IOBus Ready) : Slave>;Master, 3State … IOバススレーブは、現在のIOバスデータトランザクションが現在のクロックサイクルを最後に終了することを示すためにこの信号をアサートする。IOバスマスタは、この信号により、現在のトランザクションがこのクロックサイクルで終了することを知る。

【０１０４】b(Mastername)BReq_L (IOBus Bus Request) : Master>;Arbiter … IOバスマスタが、IOバスアービタに対し、バスの使用権要求を行う事を示す。

【０１０５】bRetry_L (IOBus Bus Retry) : Slave>;Master 3State … IOバススレーブがマスタに対し、バストランザクションの最実行を要求する。

【０１０６】b(Slavename)RdyOeReq (IOBus Slave Ready Output Enable Request) : Slave>;DefaultDriverLogic … IOバススレーブが、bRdy_L,bWBurstReq_L,bBurstAck_Lを3ステートバス上にドライブしたい場合に、IOBus Output Control Logicに対してアサートする。IOBus DefaultDriverLogicは各マスタからの、b(Slavename)RdyOeReqに基づきドライブを許すスレーブに対し、b(Slavename)RdyOe_L信号を返す。

【０１０７】b(Slavename)RdyOe_L (IOBus Slave Ready Output Enable) : DefaultDriverLogic>;Slave … b(Slavename)RdyOeReqを出力したマスタに対し、デフォルトドライバーロジックがドライブを許す場合b(Slavename)RdyOe_L信号をそのマスタに対して返す。

【０１０８】bSnoopWait (IOBus Snoop Wait) : SBB>;NextMaster … IOバスインターフェースがIOバスに接続された他のデバイスに対し、キャッシュのスヌーピング実行中であることを示す。IOバスに接続されたデバイスはこの信号がアサートされている間は新たなトランザクションを発行できない。

【０１０９】bStart_L (IOBus Transaction START) : Master >;Slave 3State … IOバスマスタがIOバストランザクションをスタートすることを示す信号、IOバススレーブは、この信号を監視することにより、IOバストランザクションのスタートを知ることが出来る。

【０１１０】bTx_L (IOBus Transaction Indicator Input) : Master>;Slave 3State …IOバスマスタがIOバススレーブに対し、IOバストランザクションが現在実行中である事を示すためにアサートする。

【０１１１】bWBurstGnt_L (IOBus Burst Write Grant) : Master>;Slave, 3State … IOバスマスタが、IOバスバーストライトのリクエストに対し、バーストライト実行することを示すためにドライブする。

【０１１２】bWBurstReq_L (IOBus Burst Write Request) : Slave >;Master, 3Stete …IOバススレーブがIOバスマスタに対し、バーストライトを要求する場合にアサートする。

【０１１３】bWr_L (IOBus Write Transaction Indicater) : Master>;Slave, 3State …IOバスマスタが、IOバススレーブに対し、現在のトランザクションがライトである事を示すためにアサートする。

【０１１４】bByteEn[3:0] (IOBus Byte Enables) : DataDriver>;DataReceiver, 3State… IOバス上にデータをドライブするエージェントが、各ビットに対応したbData[31:0]上のバイトレーンが有効である事を示すためにHighにドライブする。本信号の各ラインとbDataのバイトレーンは表4の対応関係にある。

【0115】

【表4】

Byte Enable	Corresponding bData [31 : 0]
bByte En3	[31 : 24]
bByte En2	[23 : 16]
bByte En1	[15 : 8]
bByte En0	[7 : 0]

【0116】bBurst_L (IOBus Extended Burst Request) : Master>;Slave, 3Sate … IOバスマスタが拡張バーストを行いたい事を示す。アサート、ネゲートタイミングはbTx_Lと同一。

【0117】bBurstAck_L (IOBus Extended Burst Acknowledge) : Slave>;Maste, 3State… IOバススレーブが拡張バーストを行える事を示す。アサート、ネゲートタイミングはbRdy_Lと同一。

【0118】bBurstShortNotLong_L (IOBus Burst Length) : Master>;Slave, 3State …IOバスマスタが拡張バーストを行う場合のバースト長を示す。アサート、ネゲートタイミングはbTx_Lと同一であり、信号値とバースト長との対応は表5に示したとおりである。

【0119】

【表5】

bBurst Short Not Long_L	Burst Length
H	4 beats
L	8 beats

【0120】IOバス信号は上述の通りである。DoEngine内部のバスである、IOバス（及びGバス）は接続される機能ブロック数が10以上になるので、In Out分離バスですべてのブロックを接続することは、困難である。DoEngineでは、チップ内双方向バスを採用する。

<Gバスインターフェース>図22にGバスインターフェース2006のブロック図を示す。この概要は下記の通りである。

【0121】（Gバス概要）Gバスは、MFP用1チップコントローラDoEngine内部において、各画像データ処理部間のデータ転送を高速に実行するために定義されたバスである。64ビットのデータバスをもち、4Gbyte(128byte boundary)のアドレス空間をサポートする。16ビート(128byte=64ビット x 16) を1ロングバーストとした転送を基本とし、連続して4ロングバースト(512byte=16ビート x 4) までを可能とする。（シングルビートなど16ビート以下の転送はサポートしない）

（Gバス信号定義）信号の定義に用いる記号をまず定めておく。信号名の直後には、必要に応じて信号の方向が記述されている。それは次のように定める。

In (Input signal) … バスエージェントに対する入力信号

Out (Output signal) … バスエージェントからの出力信号

InOut (Bi-Directional Tri-State signal) … 双方向の信号で、複数のバスエージェントがドライブする。一度にひとつのエージェントだけがドライブする。信号をドライブする各エージェントのイネーブルリクエスト信号をデフォルトドライバで集中管理し、どのエージェントがドライブするかはデフォルトドライバが決定する。どのエージェントもイネーブルリクエストを出さない場合や、複数のエージェントが同時にイネーブルリクエストを出している場合はデフォルトドライバ信号をドライブする。エージェントは、信号をロウにドライブする場合は、前後1クロックの間ハイにドライブしなければならない。信号のアサートはドライブを初めてから1クロック以上経過してからしか行えない。基本的に信号のリリースはネゲートした次のクロックで行う。

【0122】なお、各信号名の後の"_L"はその信号がローアクティブであることを示す。信号の記述のしかたは、ほぼIOバス信号の記述に準ずる。説明は、システム信号、アドレス及びデータ信号、インターフェース制御信号、アービトレーション信号に分けて行う。また、バスエージェントとは、バスに接続されるバスマスタやバススレーブの総称である。

【0123】（システム信号）

gClk (G-Bus Clock) … Gバス上のすべてのトランザクションについてタイミングを提供し、すべてのデバイスに対して入力となる。

【0124】gRst_L (G-Bus Reset) … Gバス上のすべてのデバイスをリセットする。すべての内部レジスタはクリアーされ、すべての出力信号はネゲートされる。

【0125】（アドレスおよびデータ信号）

gAddr[31:7], InOut, (G-Bus Address) : Master>;Slave … Gバス上のデータ転送はすべて128byte（16ビート）単位で行われるため、gAddr[31]～gAddr[7]の25ビットで4Gbyteのアドレス空間をサポートする。

drive:gTs_Lと同時にマスターがドライブ

assert:ドライブした次のクロック

negate:gAack_Lのアサートを確認したクロック、.
【０１２６】g(Mastername)AddrOeReq (G-Bus Address OutPut Enable Request) : Master>;DefaultDriverLogic … 双方向GバスをMを実現するための、デフォルトドライバロジックへの出力信号。バスマスタがアドレスバスをドライブするための要求信号。
【０１２７】g(Mastername)AddrOe_L (G-Bus Address OutPut Enablet) : DefaultDrivelogic>;Master … g(Mastername)AddrOeReqを出力したバスマスタに対し、デフォルトドライバロジックがアドレスバスのドライブを許可することを示す信号。
【０１２８】gData[63:0], InOut, (G-Bus Data) : DataDriver>;DataReceiver … 64ビットデータバスで、ライト時はマスタがドライブ、リード時はスレーブがドライブ。
【０１２９】[ライト]drive:gTs_Lと同時にマスタがドライブ。ただし、gSlvBsy_Lがアサートされている時はネゲートされるまで待ってドライブ。
assert:ドライブした次のクロック。
change:gAack_Lのアサートを確認したクロック、その後は毎クロック。
negate:転送終了時、またはgTrStp_Lによる転送停止要求を確認した場合は、gAack_Lのアサートを確認したクロック。
【０１３０】[リード]drive:gAack_Lと同時にスレーブがドライブ。
assert:スレーブがReadyであればドライブした次のクロックで、Readyでなければ Readyになるまで待ってアサートされる。
change:gAack_Lのアサートを確認したクロック、その後は毎クロック。リードの場合はアサートしたクロックから毎クロック。
negate:転送終了時、
release:ネゲートの１クロック後、またはgTrStp_Lによる転送停止要求を確認したクロック。
【０１３１】g(DataDrivername)DataOeReq (G-Bus Data OutPut Enable Request) : DataDriver>;DefaultDriverLogic … データドライバがデータバスをドライブするための要求信号。
【０１３２】g(DataDrivername)DataOe_L (G-Bus Data OutPut Enablet) : DefaultDriverLogic>;DataDriver … g(DataDrivername)DataOeReqを出力したデータドライバに対し、デフォルトドライバロジックがデータバスのドライブを許可することを示す信号。
【０１３３】(インターフェース制御信号)
gTs_L (InOut G-Bus Transaction Sart) : Master>;Slave … マスターにより１クロックの間Lowにアサートされ、転送の開始（アドレスフェーズ）をあらわす。マスターはgTs_Lと共に、gAddr,gRdNotWr,gBstCntをドライブし、転送の種類、データ量を明確にする。マスタは、ライトの場合ここで明確にした転送データ量をウェイトなしで出せることを保証しなければならない。また、リードの場合は明確にした転送データ量をウェイトなしで受けることを保証しなければならない。スレーブは途中でデータ転送ができなくなった場合はgBsStep_Lにより、次の１６ビートの転送をキャンセルすることがある。ただし、１６ビートの途中でキャンセルすることはない。
drive:gGnt_Lのアサートを確認したクロックでドライブ。
assert:ドライブした次のクロック。
negate:１クロックアサート後にネゲートされる。
【０１３４】g(Mastername)TsOeReq (G-Bus Transaction Start OutPut Enable Request) : Master>;DefaultDriverLogic … バスマスタがgTs_Lをドライブするための要求信号。
【０１３５】g(Mastername)TsOe_L (G-Bus Transaction Start OutPut Enablet) : DefaultDriverLogic>;Master … g(Mastername)TsOeReqを出力したバスマスタに対し、デフォルトドライバクロックがgTs_Lのドライブを許可することを示す信号。
【０１３６】gAack_L, InOut, (G-Bus Address Acknowledge) : Slave>;Master … スレーブにより１クロックの間Lowにドライブされる。該当するスレーブが、転送を認識し、バスが空いていることを確認して、データ転送がスタートできることをマスターに知らせる。ライトの場合、スレーブはマスタから要求された転送データ量をウェイトなしで受けることを保証しなければならない。またリードの場合は、要求された転送データ量をウェイトなしで出せることを保証しなければならない。万が一、途中でデータ転送ができなくなった場合は、gBstStp_Lにより、次の１６ビートの転送をキャンセルすることができる。ただし、１６ビートの途中でキャンセルすることはできない。
drive:アドレスデコードヒット時、gTs_Lのアサートを確認したクロックでドライブが開始される。ただし、gSlvBsy_Lがアサートされている時はネゲートされるまで待ってドライブされる。また、データバス使用中のため、まだドライブされていなかった場合は、gTrStp_Lによる転送停止要求を確認したクロックでドライブ開始される。
assert:スレーブがReadyであればドライブした次のクロックで、Readyでなければ Readyになるまで待ってアサートされる。gTrStp_Lによる転送停止に対する応答のときには、ドライブした次のクロックでアサートされる。
negate:ドライブ後にgTrStp_Lがアサートされた場合は、gTrStp_Lを確認したクロックでアサートされる。また、１クロックアサート後にネゲートされる。
【０１３７】g(Slavename)AackOeReq (G-Bus Address Acknowledge OutPut Enable Request.) : Slave>;DefaultD

riverLogic … スレーブがgAack_Lをドライブするための要求信号。

【0138】g(Slavename)AackOe_L (G-Bus Address Acknowledge OutPut Enable:) : DefaultDriverLogic>;Slave … g(Slavename)AackOeReqを出力したスレーブに対し、デフォルトドライバロジックがgAack_Lのドライブを許可することを示す信号。

【0139】gSlvBsy_L, InOut, (G-Bus Slave Busy) : Slave>;Master … スレーブがドライブし、データバスでデータを転送中であることをあらわす。

drive : アドレスデコードヒット時、gTs_Lのアサートを確認したクロックでドライブ開始。ただし、gSlvBsy_Lがアサートされている時はネゲートされるまで待ってドライブ。

assert : スレーブがReadyであればドライブした次のクロックで、Readyでなければ Readyになるまで待ってアサート。

negate : 転送終了時にネゲート。

release : ネゲートの1クロック後、またはgTrStp_Lによる転送停止要求を確認したクロック。

【0140】g(Slavename)SlvBsyOeReq (G-Bus Slave Busy OutPut Enable Request:) : Slave>;DefaultDriverLogic … スレーブがgSlvBsy_Lをドライブするための要求信号

g(Slavename)SlvBsyOe_L (G-Bus Slave Busy OutPut Enable) : DefaultDriverLogic>;Slave … g(Slavename)SlvBsyOeReqを出力したスレーブに対し、デフォルトドライバロジックがgSlvBsy_Lのドライブを許可することを示す信号。

【0141】gRdNotWr, InOut, (G-Bus Read(High)/Write(Low)) : Master>;Slave … マスターによりドライブされ、HighでREAD、LOWでWRITEをあらわす。ドライブする期間はGAと同じである。

derive : gTs_Lと同時にマスターがドライブ。

assert : ドライブした次のクロック。

negate : gAack_Lのアサートを確認したクロック。

【0142】g(Mastername)RdNotWrOeReq (G-Bus Read/Write OutPut Enable Reques) : Master>;DefaultDriverLogic … バスマスタがgRdNotWrをドライブするための要求信号。

【0143】g(Mastername)RdNotWrOe_L (G-Bus Read/Write OutPut Enable:) : DefaultDriverLogic>;Master … g(Mastername)RdNotWrOeReqを出力したバスマスタに対し、デフォルトドライバロジックがgRdNotWrのドライブを許可することを示す信号。

【0144】gBstCnt[1:0], InOut, (G-Bus Burst Counter) : Master>;Slave … マスターによりドライブされ、連続して行うバースト転送の数(1～4)をあらわす。信号の値とバースト転送するバイト数との対応は表6のようになる。

derive : gTs_Lと同時にマスターがドライブ。

assert : ドライブした次のクロック。

negate : gAack_Lのアサートを確認したクロック。

【0145】

【表6】

gBstCnt [1 : 0]		転送バイト数
01	16ビート×1	64bit × 16 × 1 = 128byte
10	16ビート×2	64bit × 16 × 2 = 256byte
11	16ビート×3	64bit × 16 × 3 = 384byte
00	16ビート×4	64bit × 16 × 4 = 512byte

【0146】g(Mastername)BstCntOeReq (G-Bus Burst Counter OutPut Enable Request:) : Master>;DefaultDriverLogic … バスマスタがgBstCntをドライブするための要求信号

【0147】g(Mastername)BstCntOe_L (G-Bus Burst Counter OutPut Enable:) : DefaultDriverLogic>;Master … g(Mastername)BstCntOeReqを出力したバスマスタに対し、デフォルトドライバロジックがgBstCntのドライブを許可することを示す信号。

【0148】gBstStp_L, InOut, (G-Bus Burst Stop) : Slave>;Master … スレーブによりドライブされ、連続する次のバースト転送を受付不可であることをあらわす。1バースト(16ビート)の転送の15ビート目にアサートする。ストップしない場合にはドライブしない。

drive : 14ビート目。

assert : 15ビート目。

negate : 1クロックアサート後

g(Slavename)BstStpOeReq (G-Bus Burst Stop OutPut Enable Request) : Slave>; DefaultDriverLogic … スレーブがgBstStp_L上をドライブするための要求信号。

【0149】g(Slavename)BstStpOe_L (G-Bus Burst Stop OutPut Enable:) : DefaultDriverlogic>;Slave … g(Slavename)BstStpOeReqを出力したスレーブに対し、デフォルトドライバロジックがgBstStp_Lのドライブを許可することを示す信号。

【0150】(アービトレーション信号)

g(Mastername)Req_L, Out, (G-Bus Request) : Master>;Arbiter … マスターによりドライブされ、アービタに対してバスを要求する。各マスターデバイスごとに専用のgReq_Lを持つ。

assert : データ転送が必要なマスターがアサート

negate：gGnt_Lを受ければネゲート
g(Mastername)Gnt_L, In, (G-Bus GNT)：Arbiter>;Master …アービタによりドライブされ、バス要求に対して次のバス権を与える。各マスタデバイスごとに専用のgGnt_Lを持つ。プライオリティーの高いバスマスタから順にバス権を与える。同じプライオリティーのマスタに対しては、バス要求のあった順番にバス権を与える。
assert：他のマスターにgGnt_Lを与えていない時、または次のクロックで他のマスタに与えていたgGnt_Lをネゲートする時に、アービトレーションによって選ばれたマスタに対してアサートする。
negate：gAack_Lのアサートを確認したクロック
gTrStp_L, In, (G-Bus Transaction Stop)：Arbiter>;Master,Slave …アービタによりドライブされ、gGnt_Lによりすでにアドレスフェーズを開始されたトランザクションを中止する。ただ、すでにgAack_Lによりデータフェーズを開始してしまったトランザクションについては中止できない。また、この信号はgAack_Lによりマスクがかけられており、gAack_Lがアサートされた時には、たとえアサートしていてもネゲートされ出力される。
assert：すでにアドレスフェーズを開始したトランザクションよりも高いプライオリティのマスタからバス要求がきた時。
negate：gAack_Lのアサートを確認したクロック
(Gバスライトサイクル) Gバスのライトサイクルは次のようになる
①マスタがバス要求、gReq_Lアサート。
②アービタが許可、gGnt_Lアサート、gReq_Lネゲート。
③gGnt_Lを受けて、マスタはgTs_L,gAddr,gRdNotWr,gBstCnt,をドライブ。
ライト動作の場合、gSlvBsy_Lがアサートされていなければ、同時にgDataもドライブする、gSlvBsy_Lがドライブされていれば、gSlvBsy_Lがフリーになるのを待ってドライブする。
④スレーブはgTs_Lがアサートされている時にアドレスをデコードし、ヒットすれば自分に対する転送であることを認識する。この時gSlvBsy_Lが他のスレーブによりアサートされていなければ、gSlvBsy_LとgAack_Lのドライブを始める。また、リードの場合はgDataもドライブする。gSlvBsy_Lが他のスレーブによりアサートされていれば、データバスは使用中であるということなので、ネゲートされるまで待ってドライブを始める。gSlvBsy_L,gAack_L,(gData)のドライブ開始後、スレーブがデータ転送の準備ができれば、それぞれの信号をアサートし、データ転送を開始する。
⑤gAack_Lがアサートされた時点でアドレスフェーズは終了し、マスタはgAddr,gRdNotWr,gBstCntをネゲートする。また、その時点からマスタはライトデータを毎クロック切り替え、gBstCntで指定されたデータ量だけ転送を行う。マスタとスレーブは、それぞれ自分でクロックをカウントして、データ転送の終了を知らなければならない。
【0151】スレーブは転送の途中でマスタから要求されたデータ転送量を転送できなくなった場合は、bStStp_Lを15ビート目でアサートすることで、次の16ビートの転送をキャンセルすることがただし、16ビートの途中でキャンセルすることはできない。
【0152】マスタおよびスレーブは、gBstStp_Lがアサートされれば、次のクロックでデータの転送を終了させなければならない。
【0153】<キャッシュインバリデーションユニット(CIU)>キャッシュインバリデーションユニット(以下CIU)2005は、IOバスからメモリへのライトトランザクションを監視し、これが起こった場合は、メモリへの書込が終了する前に、CPUシェルのキャッシュインバリデーションインターフェースを用い、CPUシェルに内蔵されたキャッシュの無効化を行う。
【0154】CPUシェルは、次の3種類の信号を使用する。
・SnoopADDR[31:5] (Cache Invalidation Address)
・DCINV (Dcache (データキャッシュ) Invalidation Strobe)
・ICINV (Icache (インストラクションキャッシュ) Invalidation Strobe)
キャッシュの無効化は最大3クロックで行われる。IOバスからメモリへのライトは3クロックで終了することはないので、キャッシュインバリデーションユニット2005は、CPUシェル401から出力されるStop_L信号を用いて無効化終了のハンドシェイクを行うことはしない。但し、将来の変更に備え、IOバス上にはbSnoopWaitをStop_Lと同一サイクルだけドライブする。
【0155】なお、現在のインプリメンテーションでは、IOバスからのライトが起こった場合は、安全のためにIcacheもインバリデーションしている。もし、OSでセルフモディファイングコードを禁止し、インストラクションとして使われる可能性のあるデータのローディング時に、意図的にインストラクションキャッシュのインバリデーションを行うなら、Icacheのインバリデーションはいらない。この場合は若干のパフォーマンスアップが望める。
【0156】<メモリマップ>図23及び図24にメモリマップを示す。図23は、左から順に、仮想メモリマップ、物理メモリマップ、Gバスのアドレス空間でのメモリマップ、IOバスのアドレス空間でのメモリマップである。また、図24は、レジスタ等を含む図23における斜線部の512メガバイトを示すマップである。
【0157】プロセッサコアのメモリモデルはR3000をベースにしている。プロセッサコアの物理アドレス空間は32ビットアドレッシングにより4Gバイトある。仮

想空間も同様に32ビットアドレッシングを行なう。ユーザープロセスの最大サイズは2Gバイトである。アドレスマッピングはカーネルモードとユーザーモードで異なる。図はMMUを使用しない場合のメモリマップである。

【0158】（ユーザーモード仮想アドレッシング）ユーザーモードの仮想アドレッシングでは2Gバイトのユーザ仮想アドレス空間(kuseg)が有効となる。このユーザーセグメントのアドレスは0x00000000から始まり、すべての有効なアクセスは0にクリアされたmsbを持つ。ユーザーモードにおいてmsbをセットしたアドレスの参照はアドレスエラー例外処理を引き起こす。TLBはkusegへのすべての参照をユーザーモードとカーネルモードで同様にマッピングする。また、キャッシャブルである。kusegは通常ユーザーコードやデータを保持するために使用される。

【0159】（カーネルモード仮想アドレッシング）カーネルモードの仮想アドレス空間には4つのアドレスセグメントを持っている。

・kuseg　仮想アドレスの0x00000000から2Gバイト。ページと単位としたキャッシングとマッピングが可能。このセグメントはカーネルメモリアクセスとユーザーメモリアクセスでオーバーラップしている。

・kseg0　仮想アドレスの0x80000000から512Mバイト。物理メモリの最初の512Mバイトにダイレクトにマップされる。参照はキャッシュされるが、アドレス変換にはTLBは使われない。kseg0は通常カーネルの実行コードやカーネルデータのために使用される。

・kseg1　仮想アドレスの0xA0000000から512Mバイト。物理メモリの最初の512Mバイトにダイレクトにマップされる　参照はキャッシュされず、アドレス変換にはTLBは使われない。kseg1は通常OSによって、I/Oレジスタ、ROMコード、ディスクバッファのために使用される。

・kseg2　仮想アドレスの0xC0000000から1Gバイト。kusegと同様、TLBにより仮想アドレスから物理アドレスにマップされる。キャッシングは自由。OSは通常kseg2をスタックやコンテキストスイッチによるリマップが必要なプロセス毎のデータのために使用する。

【0160】（仮想アドレスメモリマップ（図23（a），図2-（a））仮想アドレス空間は4Gバイトあり、システム上のすべてのメモリ、I/Oがアクセス可能である。kusegにはSYSTEM MEMORY(1GB)が存在する。

【0161】kseg0には内蔵RAM(16MB)が存在する。これは、例外処理のベクタをプログラミングしたい場合にインプリメントし、例外ベクタベースアドレスを0x80000000に設定する。このアドレスは物理アドレス空間の0x00000000にマッピングされる。

【0162】kseg1にはROM，I／O，レジスタが存在し、ブートROM(16MB)，SBB内部レジスタとMC内部レジスタ(16MB)，IOBus1/01（16MB：Gバスアービタ内部レジスタ、IOバスアービタ内部レジスタ、PMU内部レジスタなどのプリミティブなIOバスレジスタ)，IOBus1/02(16MB)，IOBusMEM(16MB)，Gbus MEM(32MB)，FONT ROM(240MB)，FONT ROM or RAM(16MB)が含まれる。

【0163】kseg2にはPCI I/O(512MB)，PCI MEM(512MB)が存在する。

【0164】kseg0，kseg1はどちらも物理アドレス空間の最初の512Mバイトにマッピングされるため、kseg0，kseg1とkusegの最初の512Mバイトはすべて同じ物理アドレス空間を参照していることになる。

【0165】（物理アドレスメモリマップ（図23（b），図24（b））物理アドレス空間も仮想アドレス空間と同様4Gバイトあり、システム上のすべてのメモリ、I/Oがアクセス可能である。

【0166】PCI I/O，PCI MEM，SYSTEM MEMORYについては仮想アドレスメモリマップと同一である。

【0167】kseg1，kseg2はどちらも物理アドレス空間の最初の512Mバイトにマッピングされるので、ROM，I/O，Regは0x00000000からの空間に存在している。

【0168】（Gバスメモリマップ（図23（c），図24（c））Gバスアドレス空間は4Gバイトあり、SYSTEM MEMORY，GBus MEM，FONTのみがアクセス可能である。

【0169】（IOバスメモリマップ（図23（d），図24（d））IOバスアドレス空間は4Gバイトあり、PCI I/O，PCI MEM，SYSTEM MEMORY，IOBus1/02，IOBus MEM，FONTのみがアクセス可能である。

【0170】IOBus 1/01はプリミティブなレジスタのため、0x1C000000から0x20000000までの空間はPCIからプロテクトがかけられており、PCIからアクセスできない。

【0171】<アドレススイッチ>アドレススイッチ2003は、Pバス，Gバス，IOバス，MCバス間のデータ転送を行なうために、マスターとなるバスからスレーブとなるバスへ、SBB402を経由してアドレス信号を送るためのものである。SBB402を経由する転送において、マスターとなり得るバスは、Pバス，Gバス，IOバスであり、スレーブとなり得るバスはIOバス，MCバスである。MCバスに対してはPバス，Gバス，IOバス，のいずれかがマスターとなり、IOバスへはPバスのみがマスターとなってアドレス信号を送る。

【0172】また、Pバス-IOバス間の転送と、Gバス-MCバス間の転送は同時に行なうことができる。

【0173】図25に、アドレススイッチ2003のブロック図を示す。スイッチシーケンサ2003aにより、スイッチ2003bを切り替えてスレーブをIOバスとMCバスとで切り換え、スイッチ2003cを切り替えてマスタをPバス，Gバス，IOバスで切り換え

る。この構成により、MCバスに対してはPバス,Gバス,IOバス,のいずれかがマスターとなり、IOバスへはPバスのみがマスターとなり、また、Pバス-IOバス間の転送と、Gバス-MCバス間の転送は同時に行なうことができる。

【0174】○データスイッチンデータスイッチはPバス,Gバス,IOバス,MCバス間のデータ転送を行なう際に,SBB内でデータの流れを切り替えるものである。ライト時はマスターからスレーブ、リード時はスレーブからマスターへデータが送られる。

【0175】図26にデータスイッチ2004のブロック図を示す。この構成において、セレクタA-1～A-3及びB-1，B-2を表7のように切り替えることで、Pバス，Gバス，IOバスのいずれかをマスタとし、IOバス，MCバスのいずれかをスレーブとして、書き込みあるいは読み出しを行うように制御できる。

【0176】

【表7】

Master	Slave	W/R	Dataの流れ	A-1	A-2	A-3	B-1	B-2
PBus	IOBus	Write	P→IO		b			
		Read	IO→P					b
	MCBus	Write	P→MC	b			a	
		Read	MC→P		a			a
Gbus	MCBus	Write	G→MC			a	b	
		Read	MC→G	a				
IOBus	MCBus	Write	IO→MC			b	b	
		Read	MC→IO		a			

【0177】○アービトレーション>SBB402内のスイッチシーケンサ2003aは、各スイッチを切り替えるにあたって、SBB外部からの以下の3種類の接続要求間のアービトレーションを行う。

1．CPU

2．Gバスバスマスタ

3．IOバスバスマスタ

このアービトレーションは、現在のバススイッチ接続状況、及び、あらかじめ設定された優先順位により決定され、その結果アドレススイッチ、データスイッチの接続が切り替えられる。

【0178】○タイミングダイアグラム>図27～図32にタイミング図を示す。図27はGバスからの書き込み 読み出しサイクルのタイミング図、図28はGバスのバースト停止サイクルのタイミング図、図29～図32は、Gバスのトランザクション停止サイクルのタイミング図である。

【0179】2．6．PCIバスインターフェース

図33は、PCIバスインターフェース416のブロック図である。

【0180】PCIバスインターフェース416は、DoEngine内部汎用IOバスであるIOバスと、チップ外部IOバスであるPCIバスの間をインターフェースするブロックである。入力ピン設定により、リセット時にPCIバスコンフィギュレーション発行可能な、ホストブリッジ構成と、PCIバスコンフィギュレーションは発行しないターゲット構成とを切り替えることができる。

【0181】IOバスインターフェースのマスタDMAコントローラ3301は、PCIバスシグナルインターフェース3302を介してPCIバスマスタからDoEngine内部のリソースにアクセス要求があった場合に、IOバスマスタとしてこのアクセス要求をIOバス内部にブリッジする。

【0182】更に、このマスタDMAコントローラ3301は、PCIバス上でマッピングされるメモリからDoEngineMemoryへのDMA転送を行う事が出来る。この際、プログラマが意図したIOバスDMAと、GバスDMAのアクセスの順番を守って動作させるために、転送先アドレス(bPciAddr[31:0])と、PCIマスターコントローラ3301のID信号(bPciID)を、IOバスとアービトレーションシーケンサに対して、バスリクエストと同時に発行する。

【0183】マスタDMAコントローラ3301はバスグラント(bPciBGnt_L)を受け取り、バスを使用しデータ転送が終了した時点で、ID信号(bPciID) のアサートを取りやめる、

【0184】なお、PCIバスは、33MHz,32ビット,PCI 2.1準拠とする。

【0185】2．7．Gバスアービタ

図34は、Gバスアービタ（GBA）406のブロック図である。

【0186】Gバスのアービトレーションは、中央アービトレーション方式であり、各バスマスタに対して専用のリクエスト信号(g(mastername)Req_L)とグラント信号(g(mastername)Gnt_L)を持つ。図34は，mastername は M1～M4となっている。Busアービタ406は、Gバストのバスマスタを4つまでサポートしており、以下のような特徴を持つ。

・アービタ内部のレジスタ3401aの設定によりアービタをプログラミングすることが出来る。レジスタ設定

はIOバスより行なう。
・すべてのバスマスタを同じ優先権として、公平にバス権を与える公平アービトレーションモードと、いずれかひとつのバスマスタの優先権を上げ、優先的にバスを使用させる優先アービトレーションモードがある。どのバスマスタに優先権を与えるかはレジスタ3401bの設定により決定される。
・優先バスマスタが連続してバスを使用することができる回数を設定することが可能。
・すでにアドレスフェーズを開始したが、データフェーズをまだ開始していないトランザクションについて、そのトランザクションをストップするためのトランザクションストップサイクルをサポート。
・複数のバスマスタにおける順序処理のプログラミングができる（後述）、プログラムされた順序は、レジスタテーブル3401aに格納される。
・Gバスマスタと IOバスマスタが同一メモリアドレスに順次書き込みを発行した場合に、プログラマの意図したアクセス順序を維持するための機構として、同期ユニットからのマスタID信号、ストップ信号に基づき、特定のマスタに対し、バス使用許可を与えることを保留する機構を持つ。
【0187】なお、レジスタへのプログラミングは、IOバスを介してCPU401から行われる。
【0188】（アービトレーションシーケンサ）Gバスアービタの中核となるアービトレーションシーケンサ3402a，bは、1つの優先マスタとその他の4つの非優先マスタの間で5つのマスタによるアービトレーションを行なう。4つのバスマスタからのリクエスト信号とグラント信号を、リクエストディスパッチ回路3403とグラントディスパッチ回路3404によって4つの非優先マスタに割り付けることにより、公平アービトレーションモードが実現される。また、4つのバスマスタのうちのひとつを、高優先アービトレーションシーケンサ3402aの優先マスタに割り付けることで、優先アービトレーションモードとして動作する。これらの割り付けは、レジスタ3401a，bの設定にしたがって行われる。これにより、優先バスマスタは、他のマスタより高い確率でバスの使用権を取得することが出来る。
【0189】さらに、バスの取得機会確率の調整に加え、高優先シーケンサ3402aに割り当てられたマスタは、連続してバスを使用する事が出来、連続して使用出来る回数をプログラマブルなレジスタにより可変する事ができる、これは、バスの占有率を調整出来、ある特定のマスタにより多くバスを使用させる様に出来ることを意味する。
【0190】（公平アービトレーションモード）このモードでは、すべてのバスマスタは同じ優先順位にあり、バス権を与えられる機会は公平である。バスがフリーの時は、一番最初にリクエストを出したバスマスタがバス権を得ることができる。また、複数のバスマスタが同時にリクエストを出した場合は、あらかじめ決められた順序にしたがって順次バス権が与えられる（ラウンドロビン方式）。例えば、M1からM4までのすべてのバスマスタが同じクロックでリクエストを出した場合は、M1→M2→M3→M4といった順序でバス権が与えられる。M4のトランザクションの終了時に再びすべてのバスマスタがリクエストを出している場合は、M1→M2→M3→M4→M1→M2…というように、同様の順序でバス権を与えていく。一部のバスマスタがリクエストを出している場合は、M4からM1へラウンドロップするとして、最後にバスを使用したマスタにもっとも近い大きい番号を持ったマスタへグラントを与える。
【0191】一度バス権が他のバスマスタに移ると、他にリクエストを出しているすべてのバスマスタにバス権を与えた後でないと再びバス権を得ることができない。
【0192】（優先アービトレーション）このモードでは、ひとつのバスマスタ（レジスタ3401bに登録されたバスマスタ）が他のバスマスタよりも高い優先権を持つ優先バスマスタになり、他のバスマスタに比べ優先的にバス権が与えられる。優先バスマスタ以外のバスマスタの優先順位はすべて同じである。
【0193】複数のバスマスタがリクエストを出しており、また優先バスマスタが連続したリクエストを行う場合、優先バスマスタと他の非優先バスマスタは交互にバス権を得る。
【0194】非優先バスマスタから他のバスマスタにバス権が移ると、他にリクエストを出しているすべてのバスマスタにバス権を与えた後でないと、その非優先バスマスタは再びバス権を得ることができない。
【0195】（トランザクション・ストップ・サイクル）優先アービトレーションモードにおいて、優先バスマスタがリクエストを出した時、すでに他のバスマスタがアドレスフェーズを開始していても、データフェーズをまだ開始していなければ、そのトランザクションをストップし、優先バスマスタがバス権を得ることができる。ただし、その直前に優先バスマスタがバス権を持っていた場合は、連続したバス権を得ることができる回数を超えることはできない。
【0196】優先バスマスタのトランザクションが終了した時、中止されたバスマスタがリクエストを出していれば、優先してバス権が与えられる。
【0197】（優先バスマスタの切り替え）優先バスマスタの切り替えを行なうには、レジスタ3401bを書き換えればよい、優先バスマスタを選択するレジスタが書き換えられると、その時に実行中のトランザクションの終了を待って、優先バスマスタが切り替えられる。アービタのステートはアイドル状態にもどり、その時点でリクエストを出していたバスマスタは、その時同時にリクエストを出したものとして、あらためてアービトレー

ションが行われる。

【0198】優先バスマスタの切り替えには十分注意を払う必要がある。優先させるべきバスマスタのDMAが終了しないうちに、異なったバスマスタに優先バスマスタを切り替えてしまうと、最初の優先バスマスタのDMAの優先度が下がってしまう。もしも最初の優先バスマスタの優先度を下げたくないのであれば、DMAが終了したのを確認してから優先バスマスタの切り替えを行なう必要がある。

【0199】優先バスマスタ切り替えを、システムブート時のみでなく、システム稼働中も動的に行う必要のあるソフトウェアでは、優先バスマスタの切り替えは、いったんGバス上に新たなDMAリクエストが発生しないよう、すべてのバスマスタ及びDMAコントローラに対する設定を中止し、その後、Gバスアービタ406内のレジスタに適切な値をセットし、さらに、Gバスアービタ406内のステータスレジスタをチェックし、バスマスタの優先権が切り替わったのを確認した上で新たなGバス上へのアクセス及びDMAの起動を行うべきである。

【0200】優先バスマスタの動的切り替えは、オペレーティングシステムの実時間保証、タスクの優先順位の設定を変化もしくは違反させてしまう可能性があり、十分な考慮の上行われなければならない。

【0201】（順序処理）図35は、DoEngine100内におけるGバス404を中心とする、Gバス上のバスマスタによるDMAに係るブロック図である。

【0202】複数のバスマスタが順次に処理を行うことが必要な場合、例えば、メモリ3501上のデータに対して、バスマスタ1により処理Aを行った後、バスマスタ2により処理Bを行い、その処理後のデータをバスマスタ4に送るといった一連の処理を考える。

【0203】この処理を行うソフトウェア、すなわちCPU401により実行されるプログラムにより、IOバス405を介して、バスアービタ406内のレジスタテーブル3401aに、バスマスタがバスを使用する順序と、バス権の付与の開始条件と、終了条件がセットされる。この例では、

バスマスタ	：	開始条件	：	終了条件
1．バスマスタ1	：	gM2BufEmpty	：	gM1BufReady
2．バスマスタ2	：	gM1BufReady	：	gM1BufEmpty
3．バスマスタ4	：	gM2BufReady	：	gM2BufEmpty

というようにセットする。すなわち、Gバスアービタ406は、それぞれのバスマスタから、開始条件として設定された信号を受けるとそのバスマスタにバス使用権を与え、終了条件として設定された信号を受けるとバス使用権を奪う

【0204】ソフトウェアはそれぞれのバスマスタにDMAをセットする。それによって、それぞれのマスタはGバスアービタ404に対してリクエスト(g(mastername)Req_L)を発行する。Gバスアービタ404はレジスタテーブル3401aに登録された順序に従ってバスマスタ1にバス権を与える(gM1Gnt_L)。バスマスタ1はメモリ3501からある単位のデータをリードし、処理Aを行って、バスマスタ1内部のバッファにデータをライトする。バスマスタ1はひとつの単位の処理が終了し、バッファの準備ができたことをgM1BufReady信号によりアービタ406に対して通知する。

【0205】アービタ406はそれを受け、レジスタテーブル3401aに登録された、バスマスタがバス権を与える条件とバス権を奪う条件にしたがって、バス権をバスマスタ1から奪いバスマスタ2に与える。バスマスタ2はバスマスタ1のバッファのデータをリードし、処理Bを行って、バスマスタ2内部のバッファにデータを格納する。この間にバスマスタ1のバッファが空になるとgM1BufEmptyがアサートされ、アービタ406はバスマスタ2にバス権を与えていたのを取りやめる。バスマスタ2は処理Bを行って、バッファの準備ができるとgM2BufReady信号により通知する。

【0206】アービタ406はそれを受け、レジスタ3101aの内容にしたがって、今度はバスマスタ4にバス権を与える。バスマスタ4はバスマスタ2のバッファからデータをリードする。バスマスタ2のバッファが空になると、gM2BufEmptyによってアービタ406に通知し、アービタ406はそれを受け、レジスタ3401aの内容に従って再びバスマスタ1にバス権を与え、次のデータの処理を始める。

【0207】それぞれのバスマスタにセットされたDMAがすべて終了すると、それぞれのバスマスタはプロセッサに割り込みによって通知する。ソフトウェアはすべてのバスマスタからの終了通知がそろったとき、一連の処理が終了したことを知る。

【0208】以上説明した動作は完全順次モードでの動作であり、順次処理に関わるバスマスタ以外のバスマスタは、バスを使用することができない。この順次処理中でも、順次処理に関係のないバスマスタがバスを使用することを可能にするために優先順次モードが用意されている。これらのモードの切り替えではアービタ406内部のレジスタにプログラミングすることによって行なう。優先順次モードでは、順次処理を行なうバスマスタは優先的にバスを使用できるが、順次処理に関わらないバスマスタであればバスを使用することが許される。順次処理を行なうバスマスタと関わらないバスマスタの間のアービトレーションは、前述した優先アービトレーションモードと同等である。当然、順次処理に係わるバスマスタで、バス権を与えられる条件が満足されずに自分の順番が回ってきていないバスマスタにはバス権は与えられない。

【0209】（アクセス順序を維持するための機構）信号stopSpecがアサートされた場合は、Gバスマスタのひ

とつであるスキャナプリンタコントローラ４０８はアービトレーションの対象から除外され、たとえリクエストをアサートしていてもバス使用権を与えられることはない。アービトレーションはこのマスタを除外したマスタ間で行われる。詳細な説明はＩＯバスアービタの節で行なう。

【０２１０】＜タイミングダイアグラム＞図３６～図３９において、Ｇバスアービトレーションのタイミングを説明する。図３６は、連続してバスを使用する回数が、バスマスタ１～４のすべてについて１に設定されている場合の公平アービトレーションモード（フェアモード）の例である。バスマスタ１による２回目の（タイミング４から出されている）バス要求は、バス要求を出している他のバスマスタがすべて１回ずつ処理されるまで待たされている。

【０２１１】図３７は、連続してバスを使用する回数が、バスマスタ１についてのみ２であり、他のバスマスタは１に設定されている場合の公平アービトレーションモードの例である。バスマスタ１が出している２回目の（タイミング４から出されている）バス要求は１回目の要求に続いて直ちに許可され、他のバスマスタはその処理がすむまで待たされている。

【０２１２】図３８は、連続してバスを使用する回数がそれぞれ１ずつであり、バスマスタ１が優先バスマスタと設定されている場合の優先アービトレーションモードの例である。優先バスマスタと非優先バスマスタとは交互にバス使用権が認められるために、バスマスタ１の２回目のバス要求はバスマスタ２のバスの使用後に認められており、バスマスタ４のバス要求は、バスマスタ１の２回目のバスの使用後に認められている。また、バスマスタ２の２回目のバス要求は、バス要求を出している他のすべてのバスマスタ、図３８ではバスマスタ１及びバスマスタ４のバス使用が終了した後で認められている。

【０２１３】図３９は、バスマスタ１からのバス要求により、バスマスタ４のバス要求が許可されているにもかかわらず中断された例である。この場合、バスマスタ１のバス使用が終了すると、バスマスタ２のバス要求に優先してバスマスタ４のバス要求が認められる。

【０２１４】２．８．ＩＯバスアービタ

図４０は、ＩＯバスアービタ４０７のブロックである。

【０２１５】ＩＯバスアービタ４０７は、ＤｏＥｎｇｉｎｅ内部のＩＯ汎用バスであるＩＯバス４０５のバス使用要求を受け付け、調停の後、使用許可を選択された一つのマスタに対して与え、同時に２つ以上のマスタがバスアクセスを行う事を禁止する。

【０２１６】アービトレーション方式は、３段階のプライオリティを持ち、それぞれのプライオリティに複数のマスタをプログラマブルに割り当てられる構成になっている。割り当ては、それぞれ最大で、最高のプライオリティに３マスタ、中間に７マスタ、最低レベルに３マスタとなる。

【０２１７】また、ＧバスマスタとＩＯバスマスタが、同一メモリアドレスに順次書き込みを発行した場合に、プログラマの意図したアクセス順序を維持するための機構として、同期ユニットからの、マスタＩＤ信号、ストップ信号に基づき、特定のマスタに対し、バス使用許可を与えることを保留する機構を持つ。

【０２１８】（アービトレーションシーケンサ）ＩＯバスアービタは３つのアービトレーションシーケンサ４００２，４００３，４００４から構成される。それぞれ、高優先権、中優先権、低優先権を持ち、それぞれのシーケンサ内に、３，７，３本のバスマスタ用アービトレーションシーケンサが内蔵される。ＩＯバス上のすべてのバスマスタになる可能性のあるユニットからのリクエスト信号とそれらへのグラント信号を、リクエストセレクタとグラントセレクタによって、これらの３つのシーケンサユニットに分配する。この分配は、ＢＢｕｓインタフェース４００５内のソフトウェアプログラマブルなレジスタ４００５ａにより、複数の組み合わせのなかから、一意の組み合わせを選択することが出来る。

【０２１９】たとえば、最大７つのマスタのリクエストを中優先アービトレーションシーケンサ４００３に接続することにより、７つのマスタ間で公平なアービトレーションが実現される。また、バスマスタのうちのいくつかを、高優先アービトレーションシーケンサ４００２に割り付けることで、これらのマスタは他のマスタより、より高い確率でバスの使用権を取得することが出来る。さらに、いくつかのリクエストを低優先シーケンサ４００４に接続することで、これらのバスの使用率を低く抑える事が出来る。さらに、バスの取得機会確率の調整に加え、高優先シーケンサ４００２に割り当てられたマスタは、連続してバスを使用する事ができ、連続して使用出来る回数をプログラマブルなレジスタ４００５ａにより可変する事ができる。これは、バスの占有率を調整でき、ある特定のマスタにより多くの時間バスを使用させる様に出来る事を意味する。

【０２２０】（公平バスアービトレーション方式）中優先シーケンサ４００３を例にとり、公平アービトレーションの実現方法を説明する。一つのシーケンサに接続されたすべてのバスマスタは同じ優先順位にあり、バス権を与えられる機会は公平である。バスがフリーの時は、一番最初にリクエストを出したバスマスタがバス権を得ることができる（ファーストカムファーストサーブ）。また、複数のバスマスタが同時にリクエストを出した場合は、あらかじめ決められた順序にしたがって順次バス権が与えられる（同時リクエスト発行時ラウンドロビン）。例えば、Ｍ１からＭ７までのすべてのバスマスタが同じクロックでリクエストを出した場合は、Ｍ１→Ｍ２→Ｍ３→Ｍ４→Ｍ５→Ｍ６→Ｍ７といった順序でバス権が与えられる。Ｍ７のトランザクションの終了時に再

びすべてのバスマスタがリクエストを出している場合は、M1→M2→M3→M4→M5→M6→M7→M1→M2というように、同様の順序でバス権を与えていく。一部のバスマスタがリクエストを出している場合は、M7からM1へラウンドラップするとして、最後にバスを使用したマスタにもっとも近い大きい番号を持ったマスタへグラントを与える。

【0221】（優先アービトレーション）IOバスインターフェースは、高優先、中優先、低優先の3つのアービトレーションシーケンサが存在する。優先順位をつけたアービトレーションは、複数のバスリクエストを選択的に、高優先、低優先、のアービタに割り振ることにより実現される。

【0222】たとえば、ひとつのマスタを高優先に割り当て、残りを中優先に割り当てることにより、ひとつのバスマスタが他のバスマスタよりも高い優先権を持つ優先バスマスタになり、他のバスマスタに比べ優先的にバス権が与えられる。同じ優先権のアービトレーションシーケンサに割り当てられたバスマスタの優先順位はすべて同じである

【0223】複数のバスマスタがリクエストを出しており、また優先バスマスタが連続してリクエストを行う場合、優先バスマスタと他のバスマスタは交互にバス権を得る。M3が優先マスタでM1，M2，M3，M4がリクエストを出し続けた場合M3→M1→M3→M2→M3→M4→M3→M1の順でバス使用権を与える。

【0224】また、高優先バスマスタは、アービタ内のプログラマブルなレジスタにあらかじめ設定された回数だけ連続したバス権を得ることができる。連続したバスを使用出来る回数は最大で4回である。

【0225】優先バスマスタ以外のバスマスタから他のバスマスタにバス権が移ると、そのバスマスタは、他にリクエストを出しているすべてのバスマスタにバス権を与えた後でないと再びバス権を得ることができない。ひとつのバスマスタが連続してリクエストを行う場合、他にリクエストを行っているバスマスタがいなければ、連続してバス権を得ることができるが、他のバスマスタがリクエストを行っていれば、そのバスマスタはあらかじめ設定された回数だけ連続してバス権を得ることができる。一度バス権が他のバスマスタに移ると、他にリクエストを出しているすべてのバスマスタにバス権を与えた後でないと再びバス権を得ることができない。

【0226】低優先アービトレーションシーケンサ4004には最大7本のリクエストを割り当てる事が出来る。低優先シーケンサ4004に割り当てられたマスタへは、中優先、高優先シーケンサに割り当てられた、すべてのマスタのリクエストがなくならないと、バスの使用権が与えられない。このシーケンサへのバスマスタの割り当ては十分な注意を持って行われなければならない

【0227】（優先バスマスタの切り替え）優先バスマスタの切り替えを行なうには、アービタ内のレジスタを書き換えればよい。優先バスマスタを選択するレジスタが書き換えられると、その時に実行中のトランザクションの終了を待って、優先バスマスタが切り替えられる。アービタのステートはアイドル状態にもどり、その時点でリクエストを出していたバスマスタは、その時同時にリクエストを出したものとして、あらためてアービトレーションが行われる。

【0228】切り替えには十分注意を払う必要がある。優先させるべきバスマスタのDMAが終了しないうちに、異なったバスマスタに優先バスマスタを切り替えてしまうと、最初の優先バスマスタのDMAの優先度が下がってしまう。もしも最初の優先バスマスタの優先度を下げたくないのであれば、DMAが終了したのを確認してから優先バスマスタの切り替えを行なう必要がある。

【0229】優先バスマスタ切り替えを、システムブート時のみでなく、システム稼働中も動的に行なう必要のあるソフトウェアでは、優先バスマスタの切り替えは、いったんIOバス上に新たなDMAリクエストが発生しないよう、すべてのバスマスタ及びDMAコントローラに対する設定を中止し、その後、IOバスアービタ407内のレジスタに適切な値をセットし、さらに、IOバスアービタ内のステータスレジスタをチェックし、バスマスタの優先権が切り替わったのを確認した上で新たなIOバス上へのアクセス及びDMAの起動を行うべきである。

【0230】優先バスマスタの動的切り替えは、オペレーティングシステムの実時間保証、タスクの優先順位の設定を変化もしくは、違反させてしまう可能性があり、十分な考慮の上行われなければならない。

【0231】（アクセス順序制御機構）IOバスアービタ407はアクセス順序制御機構を含む。アクセス順序制御機構は、同期ユニット4001と、IOバスアービタ407，Gバスアービタ406内に組み込まれたバス使用権発行抑制機構によって実現される。IOバスアービタ407内に組み込まれたバス使用権発行抑制機構は、Gバスアービタのそれと同様に動作する。つまり、stopPci信号が入力された場合は、Pciバスマスタからのバスリクエストが発行され、アービトレーションの結果、このマスタにバスの使用権を与えることが可能な状態でも、バスの使用権発行は行わず、他のマスタへバス使用権をあたえる。具体的には、stopPci信号が入力された場合は、直ちにbPciReq_Lをマスクすることにこれを行う。

【0232】LANコントローラ414からのバスリクエストおよび、ストップ信号の場合もまったく同様に動作する。図41に同期ユニット4001のブロック図を示す。同期ユニット内には複数のDMAマスタ間すべての組み合わせに関して、それぞれコンペアユニット4101～4103が接続される、DoEngineでは、Gbu

s上のDMAマスタはスキャナ／プリンタコントローラ４０８のみ存在する。ＩＯバス上には、DMAPCIユニット、LANユニットの二つが存在する。ＳＢＢ内のＩＯバスインターフェースはＩＯバス上のバスマスタであるが、メモリには直接アクセスしないので、同期ユニット４００１にはＩＤと転送先アドレスを出力しない。

【０２３３】図４２に同期ユニット内の一つコンペアユニット（Comparation Unit 1)を以下に示す。他のコンペアユニットも同様の構成を有する。

【０２３４】PCIインターフェース４１６に付属するDMAブロックもしくは、スキャナ／プリンタコントローラ４０８から、同期ユニット４００１に対し、DMAライトがプログラミングされた時点で、転送先のアドレスとそのDMAブロック固有のリクエスト信号が通知される。

【０２３５】各コンペアユニットは、各DMAブロックからリクエストが出力された時点で、アドレスを内部に持つタイマによる現在時刻とともに記憶する。次に他のDMAブロックから、DMAライトに関するアドレスとリクエストが入力された時点で、コンペアユニットは両アドレスを比較する。一致した場合にはさらに、それぞれのレジスタに格納された時間を比較し、時間的に後からDMAライトの要求を出してきたDMAブロックが接続されたバスのバスアービタに対し、このマスタに対するバス使用強化を与えないようにする。これは、stop(ID)という信号により各バスのバスアービタに通知される。

【０２３６】各バスアービタはstop(ID)信号により通知されたマスタに対しては、アービトレーションによるバス使用権の割り当てを行わない。

【０２３７】時間が経過し、先にアクセス要求を行ったバスマスタにより当該メモリアドレスへのDMAライトが終了すると、先のマスタは、同期ユニットに対するリクエストを取り下げる。同期ユニットは、２番目にDMAライトの要求を出した、DMAブロックの接続するバスのバスアービタに対し、このDMAブロックへの、バス使用権許可禁止信号の発行をとりさげる。そして、後からDMAライトを行うべきマスタのDMAライトが行われる。

【０２３８】双方のDMAライトが終了し、双方のリクエストが取り下げられると、タイマーがリセットされる。タイマーのカウントアップは再度どちらかのマスタからのリクエストが出力された時点で再度行われる。

【０２３９】２．９．スキャナ／プリンタコントローラ
図４３は、スキャナ／プリンタコントローラ４０８のブロック図である。

【０２４０】図において、スキャナ／プリンタコントローラ４０８は、Video I/Fによってスキャナおよびプリンタと接続され、内部バスGバスおよびＩＯバスにインターフェースするブロックである。大別して以下の８つのブロックから構成される。

１．スキャナ制御ユニット４３０３…ビデオI/Fを介してスキャナの動作制御を行う。

２．プリンタ制御ユニット４３０４…ビデオI/Fを介してプリンタの動作制御を行う。

３．スキャナ系画像処理ユニット４３０５…スキャナから入力される画像データに対して画像処理を行う。

４．プリンタ系画像処理ユニット４３０８…プリンタに出力する画像データに対して画像処理を行う。

５．スキャナ／ビデオ同期ユニット４３０６…スキャナから入力する画像データに対して、入力の同期タイミングを生成する。

６．プリンタ／ビデオ同期ユニット４３０７…プリンタに出力する画像データに対して、出力タイミングを生成する。プリンタとスキャナが同期可能な組み合わせの場合、スキャナ／ビデオ同期制御ユニット４３０６とともにコピー動作のビデオタイミングを生成する。

７．データ転送制御ユニット４３０２…データ転送の動作制御を行う。DMA動作ではマスターおよびスレーブの両動作をサポートする。

８．Gバス/ＩＯバス インターフェースユニット４３０１…GバスおよびＩＯバスとスキャナ、プリンタコントローラを接続するインターフェースユニットである。データ転送制御ユニット４３０２とLバスによって接続される。

【０２４１】＜スキャナ／ビデオ同期制御ユニット４３０６＞図４４にスキャナ／ビデオ同期制御ユニット４３０６のブロック図を示す。

【０２４２】（スキャナ／ビデオ同期制御ユニットの概要）スキャナ／ビデオ同期制御ユニット４３０６は、スキャナから入力される画像データの垂直同期信号（ＳＶＳＹＮＣ）、水平同期信号（ＳＨＳＹＮＣ）、画像データ同期クロック（ＳＶＣＬＫ）により、画像データの取り込みタイミング信号、画像処理のためのタイミング信号、転送バッファであるＦＩＦＯへ書き込みタイミング信号（ＳＣＦＷＲ）を生成する。

【０２４３】また、画像データの主走査方向の遅延、取り込みピクセル数、副走査方向の遅延、取り込みライン数を管理する。設定量の画像データ取込を終了したタイミングでの状態信号（ＳＡＬＬＥＮＤ）を生成する。ラインカウンタ４４０１は副走査方向の遅延と取り込みライン数を管理して、画像読み取り有効ラインの垂直同期信号（ＳＥＦＨＳＹＮＣ）を生成する。ピクセルカウンタ４４０２は主走査方向の画像取り込み遅延と取り込み画素数を管理する。取り込んだ画像データをＦＩＦＯに格納するための書き込みタイミング信号（ＳＣＦＷＲ）を生成する。ページカウンタ４４０３は入力される画像データページ単位で管理する。設定されたページ数分の画像データ入力を終了すると、終了信号（ＳＡＬＬＰＥＮＤ）を生成する。

【０２４４】ラインカウンタ４４０１．ピクセルカウンタ４４０２，ページカウンタ４４０３の各設定値はコントロールレジスタ４３１０によって読み書きされる。

・ライトデータ　　：IFWDATA[31:0]
・リードデータ　　：IFRDATA[31:0]
・ラインカウンタライト信号　：SLCSET
・ラインカウンタリード信号　：SLCRD
・ピクセルカウンタライト信号：SPCSET
・ピクセルカウンタリード信号：SPCRD
・ページカウンタライト信号　：SPAGESET
・ページカウンタリード信号　：SPAGERD

＜プリンタビデオ同期制御ユニット4307＞図45は、プリンタ　ビデオ同期制御ユニット4307のブロック図である。

【0245】（プリンタ・ビデオ同期制御ユニットの概要）プリンタ．ビデオ同期制御ユニット4307は、プリンタから入力される画像データの垂直同期信号（PVSYNC）、水平同期信号（PHSYNC）、画像データ同期クロック（PVCLK）により、画像データの取り込みタイミング信号、画像処理のためのタイミング信号、転送バッファであるFIFOからの読み出しタイミング信号（PRFRD）を生成する。

【0246】また、画像データの主走査方向の遅延、取り込みピクセル数、副走査方向の遅延、取り込みライン数を管理する。設定量の画像データ取り込みを終了したタイミングでの状態信号（PLEND）を生成する。ラインカウンタ4501は副走査方向の遅延と出力ライン数を管理して、画像出力有効ラインの垂直同期信号（PEFHSYNC）を生成する。ピクセルカウンタ4502は主走査方向の画像出力遅延と出力画素数を管理する。出力画像データをFIFOから読み出すための読み出しタイミング信号（PRFRD）を生成する。ページカウンタ4503は出力数画像データをページ単位で管理する。設定されたページ数分の画像出力が終了すると、終了信号（PALLPEND）を生成する。

【0247】ラインカウンタ4501，ピクセルカウンタ4502，ページカウンタ4503の各設定値はコントロールレジスタ4310によって読み書きされる。

・ライトデータ　　：IFWDATA[31:0]
・リードデータ　　：IFRDATA[31:0]
・ラインカウンタライト信号　：PLCSET
・ラインカウンタリード信号　：PLCRD
・ピクセルカウンタライト信号　：PPCSET
・ピクセルカウンタリード信号　：PPCRD
・ページカウンタライト信号　：PPAGESET
・ページカウンタリード信号　：PPAGERD

＜スキャナFIFOコントローラ4311＞図46は、スキャナFIFOコントローラのブロック図である。スキャナFIFOコントローラ4312（図43参照）は、スキャナから入力された画像データをGバスもしくはIOバスを介して転送するためのバッファとしてのFIFOと、そのFIFOをコントロールする回路を含む。容量1024バイト、データ幅24ビット（R，G，B各8ビット）の2系統のFIFO4602，4603を備える。これらのFIFOは、Gバス/IOバスのデータ転送動作とスキャナからの画像データ入力動作を交互に行うダブルバッファとして動作する。FIFOのデータ入出力は、スキャナFIFOセレクタ4601がFIFOのフルフラグ（FF）とエンプティーフラグ（EF）により制御する。

【0248】＜プリンタFIFOコントローラ4312＞図47は、プリンタFIFOコントローラ4312（図43）のブロック図である。プリンタFIFOコントローラ4312は、プリンタへ出力する画像データをGバスもしくはIOバスを介して転送するためのバッファとしてのFIFOと、そのFIFOをコントロールする回路を含む。容量1024バイト、データ幅24ビット（R，G，B各8ビット）のFIFOを2系統備える。これらは、Gバス/IOバスのデータ転送動作とプリンタへの画像データ出力動作を交互に行うダブルバッファとして動作する。FIFOのデータ入出力は、プリンタFIFOセレクタ4701がFIFOのフルフラグ（FF）とエンプティーフラグ（EF）により制御する。

【0249】スキャナプリンタコントローラ408は、スキャナとプリンタとを同期動作させ、コピー動作をおこなうために、図43に示すように、スキャナデータを直接プリンタへ出力するデータバスを備える。また、スキャナからの入力データとプリンタへの出力データをFIFOにより同期させるデータバスも備える。それらのデータバスはスキャナとプリンタのデータ入出力の同期関係で選択可能とする。

【0250】＜データ転送制御ユニット4302＞図48は、データ転送制御ユニット4302のブロック図である．データ転送制御ユニット4302は、IOバスへのデータの入出力を制御するブロックで、チェインコントローラ4801、マスタDMAコントローラ4802、転送アービタ4803等を含む。

【0251】データ転送制御ユニットは．マスタとしては以下の動作を制御する。

1．スキャナからの画像データDMA転送およびチェインテーブル参照
2．プリンタへの画像データDMA転送およびチェインテーブル参照

スレーブとしては以下の動作を制御する。
1．内部レジスタのライト・リード
2．スキャナからの画像データ転送
3．プリンタへの画像データ転送
（チェインコントローラ）図49は、チェインコントローラ4801のブロック図である。チェインコントローラは、チェインテーブルを示すアドレスポインタブロックとDMAの転送先を示すアドレスポインタブロックから構成される。スキャナ、プリンタコントローラがマスタとしてDMAを行う際の転送先アドレス発生ブロックであり、チェインDMAをサポートする。スキャナデータ転送、プリンタデータ転送の2系統が独立で動作する。

【0252】・Lバス>Gバス/IOバスインターフェースユニットとデータ転送ユニットを接続するスキャナ、プリンタコントローラ内のローカルなバスである。次のような信号を含む。なお、以下、信号の入出力は、データ転送制御ユニット4302からGバス/IOバスインターフェースユニット4301に出力する信号をOUT、Gバス/IOバスインターフェースユニット4301からデータ転送制御ユニット4302に入力する信号をINで示す。

・IFCLK（IN）…　Lバスの基本クロック

・IFRDATA[63:0]（OUT）…　データ転送制御ユニットからGバス/IOバスインターフェースユニットに出力される64ビットのデータバスである。データ転送制御ユニットがマスタとスレーブのどちらの動作においても使用される。

・IFWDATA[63:0]（IN）…　Gバス/IOバスインターフェースユニット からデータ転送制御ユニットに出力される64ビットのデータバスである。データ転送制御ユニットがマスタとスレーブのどちらの動作においても使用される。

・IFMDTREQ（OUT）…　データ転送制御ユニットがマスタとしてデータ転送の要求とデータおよびアドレスの有効状態を示す。"High"でデータの転送要求とアドレスバスIFMAD[3:12]の有効状態を示す。

・IFMAD[31:2]（OUT）…　データ転送制御ユニットがマスタ動作時、目的アドレスを示すアドレスバス、IFMDTREQ信号がアクティブ状態"High"で有効なアドレスを出力する。

・IFMRW（OUT）…　データ転送制御ユニットがマスタ動作時、データの入出力を示す信号。"High"のときデータ転送制御ユニットはIFWDATA[63:0]からデータを入力する。"Low"のときデータ転送制御ユニットはIFRDATA[63:0]にデータを出力する。

・IFMDTACK（IN）…　データ転送制御ユニットがマスタ動作時、IFMDTREQに対する応答信号、Gバス/IOバスインターフェースユニットから出力される。"High"のときIFRDATA[63:0]またはIFWDATA[63:0]が有効データであることを示す。

・PRIOR[3:0]（OUT）…　データ転送制御ユニットがマスタ動作時、データ転送の優先度を示す。優先度は表8のように定義される。

【0253】

【表8】

PRIOR3	PRIOR2	PRIOR1	PRIOR0	優先度
1	1	1	1	高い
1	1	1	0	↑
		:		
0	0	0	1	↓
0	0	0	0	低い

【0254】・MTSIZE[2:0]（OUT）…　データ転送制御ユニットがマスタ動作時、転送の単位サイズを示す。この単位サイズの転送中、IFMDTREQは"High"のアクティブ状態のままとなる。サイズは表9のように示される。

【0255】

【表9】

MSIZE2	MSIZE1	MSIZE0	転送単位サイズ
1	1	1	64bit×64
1	1	0	64bit×48
1	0	1	64bit×32
1	0	0	64bit×16
0	1	1	64bit×4
0	1	0	64bit×2
0	0	1	64bit
0	0	0	32bit

【0256】・IFSDTREQ（IN）…　データ転送制御ユニットがスレーブ動作時、Gバス/IOバスインターフェ

ースユニットからのデータ転送要求とデータおよびアドレスの有効状態をしめす。"High"でデータ転送要求とアドレスバスIFSAD[6:2]の有効状態を示す。

・IFSAD[6:2]（IN）…　データ転送制御ユニットがスレーブ動作時、目的アドレスを示すアドレスバス。IFSDTREQ信号がアクティブ状態"HIGH"で有効なアドレスを出力する

・IFSRW（IN）…　データ転送制御ユニットがスレーブ動作時、データの入出力を示す信号。"High"のときデータ転送制御ユニットはIFRDATA[63:0]にデータを出力する。"Low"のときデータ転送制御ユニットはIFWDATA[63:0]からデータを入力する。

・IFSDTACK（OUT）…　データ転送制御ユニットがスレーブ動作時、IFSDTREQに対する応答信号。データ転送制御ユニットから出力される。"High"のときIFRDATA[63:0]またはIFWDATA[63:0]が有効データであることを示す。

・STSIZE（IN）…　データ転送制御ユニットがスレーブ動作時、Gバス/ＩＯバスインターフェースユニットからのデータ幅を示す。"High"でIFRDATA[63:0]またはIFWDATA[63:0]が有効となる。"Low"でIFRDATA[31:0]またはIFWDATA[31:0]が有効となる。

【０２５７】Gバス/ＩＯバスインターフェースユニットの図５０は、Gバス/ＩＯバスインターフェースユニット４３０１のブロック図である。Gバス/ＩＯバスインターフェースユニットは、コピーエンジン内部の内部バス（Ｌバス）と外部バス(Gバス/ＩＯバス)とのインターフェース部分である。このユニットは、他の機能ブロックで使えるように実現されるが、本実施の形態ではコピーエンジンが使用する。なお、コピーエンジンとは、スキャナ　プリンタコントローラ４０８及びそれにより制御されるスキャナとプリンタとを含めた総称である。

【０２５８】Gバス/ＩＯバスインターフェースユニットは、大きくバスセレクタユニット，ＩＯバスコントローラ５００２，Gバスコントローラ５００３の三つの部分から構成される。

【０２５９】バスセレクタユニット５００１は、コピーエンジンがDMAマスタの時には、Ｌバスのバースト転送可能な転送量、優先度（緊急度）、転送先のアドレスと、さらに各バス(GバスとＩＯバス）の空き情報を基にして、動的にバスの選択を行い、対応するコントローラ（Gバスコントローラ５００３かＩＯバスコントローラ５００２）に、多少の前処理を施してＬバスを接続する。コピーエンジンがDMAスレーブの時には、各バス（GバスとＩＯバス）からの要求を調停し、優先順位の高いバスとＬバスを接続する。

【０２６０】Gバスコントローラ５００３とＩＯバスコントローラ５００２は、Ｌバスと対応するバス（GバスとＩＯバス）のバスの変換を行う。以下に、各部の説明をする。

【０２６１】コピーエンジンは、Gバス/ＩＯバスの両方のバスに対してDMA可能なDMAマスタであり、Gバス/ＩＯバスインターフェースユニット４３０１は、DMA転送する際に使用するバスを決定する。従来のシステムでは、転送先（アドレス）によりバスを切り替えているが、各バスの転送速度や使用率等を考慮しなければ、システム全体として良い性能が得られない。

【０２６２】バスセレクタユニット５００１は、Ｌバスのバースト転送可能な転送量、優先度（緊張度）、転送先のアドレスと、さらに各バス（GバスとＩＯバス）の空き情報を基にして、動的に効率的なバスの選択を行い、対応するコントローラ(Gバスコントローラ５００３かＩＯバスコントローラ５００２）に、多少の前処理を施して、Ｌバスを接続する。

【０２６３】Gバスコントローラ５００３とＩＯバスコントローラ５００２は、Ｌバスと対応するバス（GバスとＩＯバス）同期ユニットへ、書き込みアドレスと機能ブロック（ここでは、コピーエンジン）のＩＤを送り出す。以下に、各部の説明をする。

【０２６４】（バスセレクタユニット）図５１は、バスセレクタユニット５００１のブロック図である。その動作を以下に説明する。

［コピーエンジンのマスター時の動作］コピーエンジンがマスターの場合は、主に、バスセレクタユニットのＬバスマスタシーケンサ５１０１で制御される。Ｌバスマスタシーケンサ５１０１は、コピーエンジンからIFMDTREQ（マスターデータ要求信号）を受けることにより、コピーエンジンのマスター動作の要求を知る。

【０２６５】コピーエンジンは、IFMDTREQアサートと同時に、IFMAD[31:2]（マスター転送アドレス信号）、MTSIZE[2:0]（マスター転送長信号）、IFMRW（マスター・リード・ライト信号）をバスセレクタユニット５００１に出力する。転送アドレスはアドレスカウンタ５１０２に、転送長は長さカウンタ５１０３に、それぞれラッチされる。

【０２６６】Ｌバスマスタシーケンサ５１０１は、外部バストの転送を開始する時点でGバスを使用するか、ＩＯバスを使用するかを、アドレスカウンタ，長さカウンタ，プライオリティ，各バスのビジー状態から決定する。アドレスカウンタの下位ビット５がすべて０ではない場合、あるいは長さカウンタが６４ビット×４未満の場合は、Gバスでの転送は不可能なので、ＩＯバスを選択する。それ以外は、高プライオリティかつGバスが使用中かつＩＯバスが空いている、という場合を除き、Gバスを選択する。外部バスへの転送サイクルを終了した時点で、アドレスカウンタ５１０２と長さカウンタ５１０３を更新し、長さカウンタが０でなければ、上記の動作を繰り返す。バスの選択の基準を表１０に示す。

【０２６７】

【表10】

アドレスカウンタ[4:0]＝0 & 長さカウンタ≧64bit×4	プライオリティ	バスの状態		選択するバス
		IOバス	Gバス	
No	—	—	—	IOバス
Yes	低い	—	—	Gバス
	高い	Ready	Ready	Gバス
		Ready	Busy	IOバス
		Busy	Ready	Gバス
		Busy	Busy	Gバス

【0268】コピーエンジンから外部バスへの転送では、Lバスマスタシーケンサ5101は、データFIFO5104がフルでない限り、コピーエンジンへIFMDTACK信号をアサートし、転送データを要求し、得たデータをFIFO5104に書き込む。また、データFIFOがエンプティでない限り、外部バスコントローラ（IOバスコントローラ5002かGバスコントローラ5003）へマスター転送要求信号（LbMReqかLgMReq）をアサートし、データ転送を要求する。外部バスコントローラ（IOバスコントローラかGバスコントローラ）は、データFIFO5104のデータを転送し、転送終了で、マスター転送通知信号（LbMAckかLgMAck）をアサートするので、Lバスマスタシーケンサ5101は、外部バスへの転送の終了を知ることができる。

【0269】外部バスからコピーエンジンへの転送では、Lバスマスタシーケンサ5101は、データFIFOがフルでない限り、外部バスコントローラ（IOバスコントローラかGバスコントローラ）へマスター転送要求信号（LbMReqかLgMReq）をアサートし、データ転送を要求し、データをデータFIFOに書き込み、データFIFOがエンプティでない限り、コピーエンジンへIFMDTACK信号をアサートし、転送データを要求し、得たデータFIFOのデータを書き込む。

【0270】（IOバスコントローラ）図52は、IOバスコントローラ5002のブロック図である。

【0271】IOバスコントローラ5002は、LバスとIOバスのインターフェース部分である。

【0272】IOバスマスタシーケンサ5201は、IOバスマスター時の動作を、IOバススレーブシーケンサ5202は、IOバススレーブの時の動作を、それぞれ制御する。

［コピーエンジンのマスター時の動作］バスセレクタユニット5001からのLbMReq信号のアサートにより、データ転送は開始される。転送の方向は、LbMReq信号と同時にアサートされるバスセレクタユニット5001からのLbMRdNotWr信号により決まる。転送のサイズは、LbMReq信号と同時にアサートされるバスセレクタユニット5001からのLbBstCnt[1:0]信号により決まる。また、転送のアドレスはLbMReq信号と同時にアサートされるバスセレクタユニット5001からのLbMAddr[31:2]信号により決まる。

【0273】IOバスからの転送(LbMRdNotWr信号が0の時）は、読み込み用データFIFOがフルであれば（bRFifoFullがアサート）、フルでなくなるのを待ち、IOバスマスタシーケンサは、決められたIOバスの転送を開始し、得たデータは、バスセレクタユニット5001 のデータFIFOに書き込む。

【0274】IOバスへの転送（LbMRdNotWr信号が1の時）は、書き込み用データFIFOがエンプティであれば（bWFifoEmplがアサート）、エンプティでなくなるのを待ち、IOバスマスタシーケンサは、決められたIOバスの転送を開始し、バスセレクタユニット5001のデータFIFOから得たデータをIOバス上に送る。

【0275】（Gバスコントローラ）図53は、Gバスコントローラ5003のブロック図である。

【0276】Gバスコントローラ5003は、LバスとGバスのインターフェース部分である。

【0277】Gバスマスタシーケンサ5301は、Gバスマスター時の動作を、Gバススレーブシーケンサ5302は、Gバススレーブの時の動作を、それぞれ制御する。

［コピーエンジンのマスター時の動作］バスセレクタユニット5001からのLbMReq信号のアサートにより、データ転送は開始される。転送の方向は、LbMReq信号と同時にアサートされるバスセレクタユニット5001からのLbMRdNotWr信号により決まる。転送のサイズは、LbMReq信号と同時にアサートされるバスセレクタユニット5001からのLbBstCnt[1:0]信号により決まる。また、転送のアドレスはLbMReq信号と同時にアサートされるバスセレクタユニット5001からのLbMAddr[31:2]信号により決まる。

【0278】Gバスからの転送(LgMRdNotWr信号が0の

時）は、読み込み用データＦＩＦＯがフルであれば（gRFifoFullがアサート）フルでなくなるのを待ち、Ｇバスマスタシーケンサは、決められたＧバスの転送を開始し、得たデータは、バスセレクタユニット５００１のデータＦＩＦＯに書き込む。

【０２７９】Ｇバスへの転送（LgMstNotWr信号が１の時）は、書き込み用データＦＩＦＯがエンプティであれば（bWFifoEmptがアサート）、エンプティでなくなるのを待ち、Ｇバスマスタシーケンサは、決められたＧバスの転送を開始し、バスセレクタユニット５００１のデータＦＩＦＯから得たデータをＧバス上に送る。

【０２８０】２．１０．電力管理ユニット

図５４は、電力管理ユニット４０９のブロック図である。

【０２８１】ＤｏＥｎｇｉｎｅはＣＰＵを内蔵した大規模なＡＳＩＣである。このため、内部のロジックが全部同時に動作してしまうと、大量の熱を発生し、チップ自体が破壊されてしまう恐れがある。これを防ぐために、ＤｏＥｎｇｉｎｅは、ブロック毎の電力の管理、すなわちパワーマネジメントを行ない、更にチップ全体の消費電力量の監視を行なう。

【０２８２】パワーマネージメントは、それぞれのブロックが各自個別に行なう。各ブロックの消費電力量の情報は、パワーマネージメントレベルとして、電力管理ユニット（ＰＭＵ）４０９に集められる。ＰＭＵ４０９では、各ブロックの消費電力量を合計し、その値が限界消費電力を超えないように、ＤｏＥｎｇｉｎｅの各ブロックの消費電力量を一括して監視する。

【０２８３】<動作>電力管理ブロックの動作は次の通りである。

・各ブロックは、４段階のパワーマネジメントレベルを持つ

・ＰＭＵは、それぞれのレベルにおける消費電力の値をレジスタとして持つ。このレベル構成及び消費電力の値は、ＰＭ構成レジスタ５４０１に保持される。

・ＰＭＵは、各ブロックからパワーマネジメントレベルを２ビットのステータス信号として受け取り（後述）、レジスタ５４０１に設定された値と照合させて各ブロックの消費電力を知る。

・ＰＭＵは各ブロックの消費電力を加算器５４０３で加算し、ＤｏＥｎｇｉｎｅ全体の消費電力量をリアルタイムに計算する。

・算出された消費電力量は、構成レジスタ５４０１に設定された消費電力のリミット値（ＰＭリミット）と比較器５４０４で比較され、これを超えた場合には割込発生器５４０５から割り込み信号を発行する。

・このリミット値は２段階設定することができる。１段階目は、本当の限界から少し余裕を持たせた値を設定しておく。この値を超えると、通常の割込み信号が発行される。ソフトウェアはこれを受けて、新たにブロックを起動するような転送は始めないようにする。ただ、２段階目のリミット値に達しない範囲内では、ソフトウェアの管理の元で、新たなブロックを起動することができる。２段階目のリミット値は、デバイスが破壊される恐れのある値を設定しておく。万が一、この値を超えてしまった時には、ＮＭＩ（割込マスクが設定できない割り込み）を発行して安全のためにシステムを停止させる。

・割込み信号は、ＰＭＵの状態レジスタ５４０２をリードすることで解除される。この状態レジスタ５４０２をリードした時点からタイマカウントを始め、タイマがエクスパイアするまで消費電力量が戻らなければ、再び割込み信号を発行する。このタイマの値の設定はＰＭＵの構成レジスタ５４０１に設定される。

【０２８４】<各ブロックのパワーマネージメント>各ブロックのパワーマネージメント制御は、ブロック毎に自由に構成してよい。構成例の例を示す。

【０２８５】（構成例１）この例では、内部ロジックへのクロックをオン／オフすることにより、パワーマネジメントを行なっているので、消費電力のレベルは２段階しか持っていない。このレベルをステータス信号として電力管理ユニット４０９に送る。図５５にバスエージェントのブロック図を示す。

・バスエージェント５５０１は、各ユニット毎の内部ロジック５５０２、アドレスをデコードするデコーダ５５０３、クロックコントロール部５５０４、クロックゲート５５０５を含む。

・デコーダ５５０３とクロックコントロール部５５０４は常に動作しており、パワーマネージメントコントロールとして、バスのアクティビティーの監視、内部ロジックへのクロックのゲーティングを行なっている。

【０２８６】<クロックコントロール>

・バスエージェントは、バスのアクティビティーを検出し、クロックのオン／オフを自動で行なう。

・バスエージェントには、Sleep, Wake Up, Waitの３つのステートがある。

・Sleepはバスエージェントにアクティビティーがなく、クロックゲートクロックを停止させている状態である。

・Sleepの状態でも、デコーダ部５５０３、クロックコントロール部５５０４は動作しており、バスをモニタリングして、要求を待っている。

・デコーダ５５０３が自分のアドレスを検出すると、クロックゲート５５０５を開き、内部ロジックのクロックを動作させ、バスの要求に応える。ステートはWake Upに移行する。また、この状態を電力管理ユニット４０９に通知する。

・データ転送が終了すると、Waitステートに移行し、次の要求を待つ。この時クロックは動作したままである。要求があれば、Wake Upステートに戻り、転送を行なう。また、要求を待っている間はタイマによりカウント

を行ない、要求がないままカウンタがタイムアップした場合は、Sleepステートへ移り、クロックを停止させる。この状態も電力管理ユニット409に通知される。

【0287】以上のようにして、消費電力が所定値を越えないように管理している。

[その他の構成例] 図9及び図10で示したキャッシュの動作手順は、図56及び図57のようなものであってもよい。

【0288】図56においては、MCバスよりデータ転送が開始されると、その最初にMCバスで示されるmTType[60:0]によってその転送をキャッシュオンで行うかオフで行うか判断される。ここでは、転送がバースト転送の時、そのバースト転送データ量がキャッシュの1ラインのデータ量より大か小かによって判断する。なお、キャッシュの1ラインは256ビット=4バースト分である。

【0289】図56において、メモリコントローラは転送の開始でmTType[3:0]をチェックし、mTTypeが示すバースト長が1/2/4であれば、キャッシュオンで動作し、6/8/16/2×16/3×16/4×16の場合はキャッシュオフで動作する。キャッシュオンあるいはオフ後の動作は、図9及び図10と同様である。

【0290】また、図58及び図59のように、デバイスによってキャッシュのオンオフを切り替えることもできる。図58において、メモリコントローラは転送の開始でmTType[6:4]をチェックすることにより転送要求デバイスを識別し、そのデバイスの転送要求をキャッシュオンで動作するか、オフで動作するかを判断するために、あらかじめ設定されている構成レジスタの値を参照し、キャッシュオンで動作するか、オフで動作するか決定する。キャッシュオンあるいはオフ後の動作は、図9及び図10と同様である。構成レジスタの設定は、ハードウエア的に決定（変更不可）していも良いし、ソフトウエアで書換え可能にしてもよい。

【0291】

【発明の効果】以上説明したように、本発明では、DMAを連続して行う場合に、ソフトウェアがバスマスタが変わる都度介入するのではなく、最初に一括してソフトウェアがバスアービタにDMAの開始させる条件と終了させる条件とを設定しておき、バスマスタにもDMAの設定を行っておくことで、後はバスアービタが順序をコントロールしながら一連の処理を行うことができる。これにより、処理毎のソフトウェアの介入が必要なくなる。また、いちいちメモリに書き戻す必要がないため、データがバスを使用する回数が少なくなり、全体としての処理速度が向上する。

【0292】また、クロスバススイッチにより要求に応じて接続するバスを切り替え、最適なバスを選んでデータの転送を行うことができる。さらに、バスの構成を柔軟にし、更にマスタとスレーブとが重複しないのであれば、複数のバスの接続を並列に行えるようにして、バスの使用効率を向上させることができる。

【0293】また、複数のバスのそれぞれに接続されたバスマスタについて、それらがバスの使用権を得た順にメモリにアクセスするように制御することができる。そのため、時間経過に伴う処理順序を適正に保つことができる。

【0294】また、転送速度が高速なバーストモードではキャッシュを用いず、シングルモードでキャッシュを使用するため、キャッシュされた大量のデータが無駄になる事態を防止してキャッシュの使用効率を向上させ、メモリへのデータ転送速度が向上する。

【0295】また、各バスマスタが使用するバスを、バスの使用状況やバス要求の優先度、バスの性能、転送するデータがバスに適合するか否かなどを動的に決定して選択するため、バスの効率を向上させることができる。

【0296】また、回路の各ブロックの動作状態を監視して消費電力を抑制し、さらに、予備的な警告と危険な事態とで別々な割込み信号で通報するため、大量の熱の発生を抑制し、装置の熱による破壊を防止することができる。

【0297】

【図面の簡単な説明】

【図1】DoEngineを用いた装置あるいはシステムの構成例の図である。

【図2】DoEngineを用いた装置あるいはシステムの構成例の図である。

【図3】DoEngineを用いた装置あるいはシステムの構成例の図である。

【図4】DoEngineのブロック図である

【図5】キャッシュメモリコントローラの3つのステートを示す図である。

【図6】インタラプトコントローラ410のブロック図である。

【図7】メモリコントローラ403のブロック図である。

【図8】キャッシュコントローラ706を中心とする詳細なブロック図である。

【図9】MCバスよりメモリリード/ライト転送が要求された場合のキャッシュの動作を示すフローチャートである。

【図10】MCバスよりメモリリード/ライト転送が要求された場合のキャッシュの動作を示すフローチャートである。

【図11】ROM/RAMコントローラ707の構成を示す図である。

【図12】CPUからのバースト読み出しのタイミングを示すタイミング図である。

【図13】CPUからのバースト書き込みのタイミング

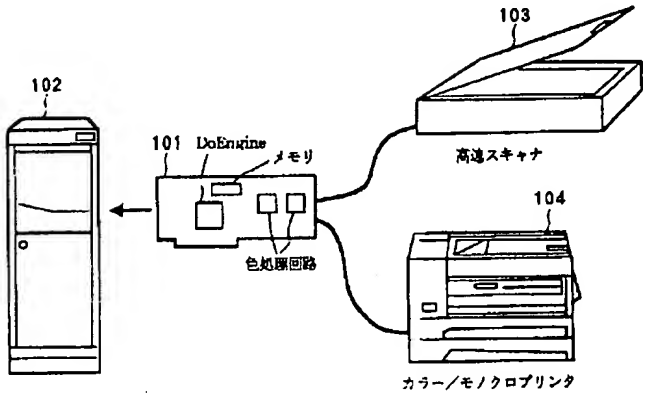
を示すタイミング図である。
【図14】Gバスデバイスからのバースト読み出しのタイミングを示すタイミング図である。
【図15】Gバスデバイスからのバースト書き込みのタイミングを示すタイミング図である。
【図16】メモリフロントキャッシュにヒットした場合のシングル読み出しのタイミングを示すタイミング図である。
【図17】メモリフロントキャッシュにヒットしなかった場合のシングル読み出しのタイミングを示すタイミング図である。
【図18】メモリフロントキャッシュにヒットした場合のシングル書き込みのタイミングを示すタイミング図である。
【図19】メモリフロントキャッシュにヒットしなかった場合のシングル書き込みのタイミングを示すタイミング図である。
【図20】システムバスブリッジ（SBB）402のブロック図である。
【図21】IOバスインターフェースのブロック図である。
【図22】Gバスインターフェース2006のブロック図である。
【図23】仮想メモリマップ、物理メモリマップ、Gバスのアドレス空間でのメモリマップ、IOバスのアドレス空間でのメモリマップの図である。
【図24】レジスタ等を含む図23における斜線部の512メガバイトを示すマップの図である。
【図25】アドレススイッチ2003のブロック図である。
【図26】データスイッチ2004のブロック図である。
【図27】Gバスからの書き込み／読み出しサイクルのタイミング図である。
【図28】Gバスのバースト停止サイクルのタイミング図である。
【図29】Gバスのトランザクション停止サイクルのタイミング図である。
【図30】Gバスのトランザクション停止サイクルのタイミング図である。
【図31】Gバスのトランザクション停止サイクルのタイミング図である。
【図32】Gバスのトランザクション停止サイクルのタイミング図である。
【図33】PCIバスインターフェース416のブロック図である。
【図34】Gバスアービタ（GBA）406のブロック図である。
【図35】DuEngine400内におけるGバス404を中心とする、Gバス上のバスマスタによるDMAに係るブロック図である。
【図36】連続してバスを使用する回数が、バスマスタ1～4のすべてについて1に設定されている場合の公平アービトレーションモード（フェアモード）の例を示す図である。
【図37】連続してバスを使用する回数が、バスマスタ1についてのみ2であり、他のバスマスタは1に設定されている場合の公平アービトレーションモードの例を示す図である。
【図38】連続してバスを使用する回数がそれぞれ1ずつであり、バスマスタ1が優先バスマスタと設定されている場合の優先アービトレーションモードの例を示す図である。
【図39】バスマスタ1からのバス要求により、バスマスタ4のバス要求が許可されているにもかかわらず中断された例を示す図である。
【図40】IOバスアービタ407のブロック図である。
【図41】同期ユニット4001のブロック図である。
【図42】同期ユニット内の一つコンペアユニットの図である。
【図43】スキャナ／プリンタコントローラ408のブロック図である。
【図44】スキャナ／ビデオ同期制御ユニット4306のブロック図である。
【図45】プリンタ／ビデオ同期制御ユニット4307のブロック図である。
【図46】スキャナFIFOコントローラのブロック図である。
【図47】プリンタFIFOコントローラ4312のブロック図である。
【図48】データ転送制御ユニット4302のブロック図である。
【図49】チェインコントローラ4801のブロック図である。
【図50】Gバス/IOバスインターフェースユニット4301のブロック図である。
【図51】バスセレクタユニット5001のブロック図である。
【図52】IOバスコントローラ5002のブロック図である。
【図53】Gバスコントローラ5003のブロック図である。
【図54】電力管理ユニット409のブロック図である。
【図55】バスエージェントのブロック図である。
【図56】MCバスよりメモリリード／ライト転送が要求された場合のキャッシュの動作の他の例を示すフローチャートである。
【図57】MCバスよりメモリリード／ライト転送が要

求された場合のキャッシュの動作の他の例を示すフローチャートである。

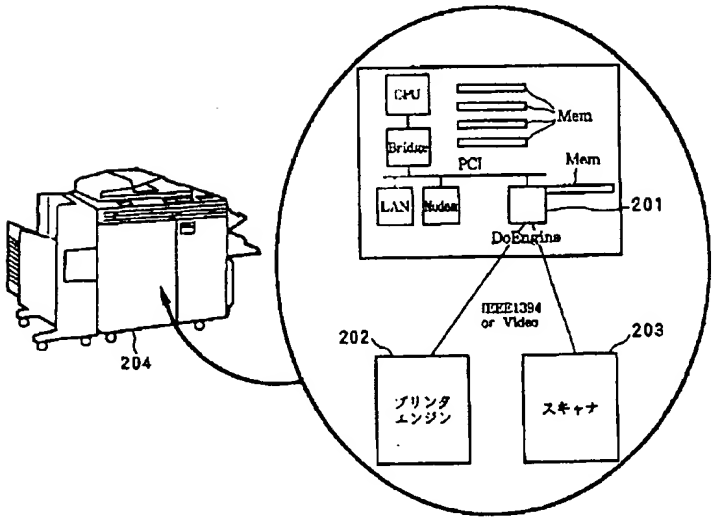
【図58】MCバスよりメモリリード／ライト転送が要求された場合のキャッシュの動作の他の例を示すフローチャートである。

【図59】MCバスよりメモリリード／ライト転送が要求された場合のキャッシュの動作の他の例を示すフローチャートである。

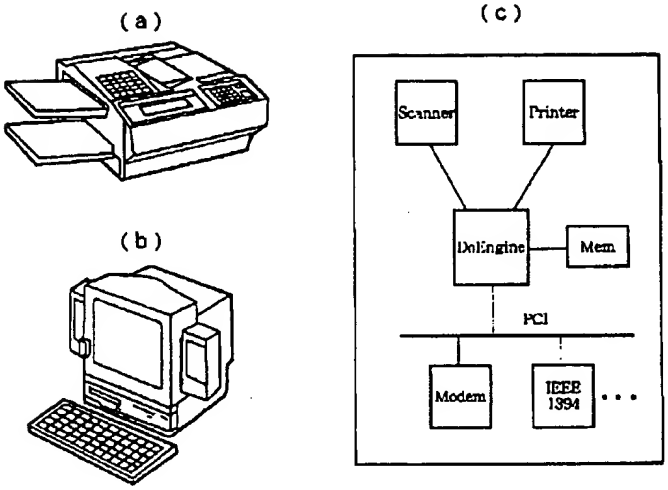
【図1】



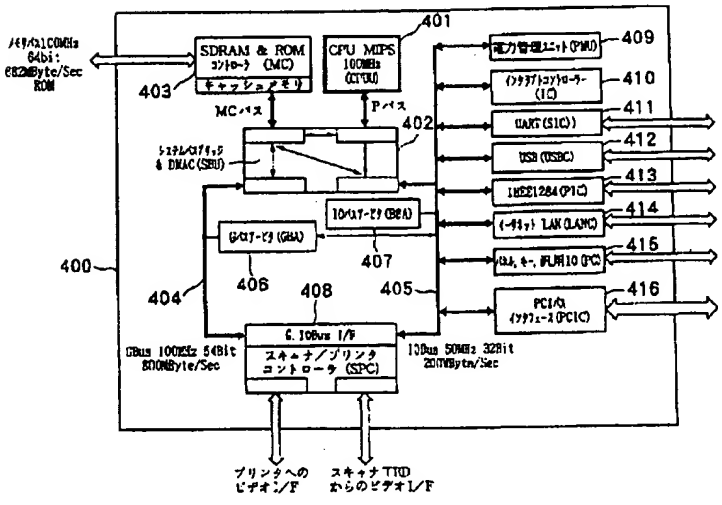
【図2】



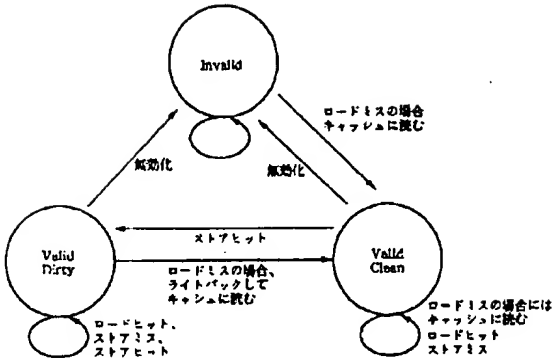
【図3】


(a)
(b)
(c)
Scanner
Printer
DnEngine
Mem
PCI
Modem
IEEE 1394

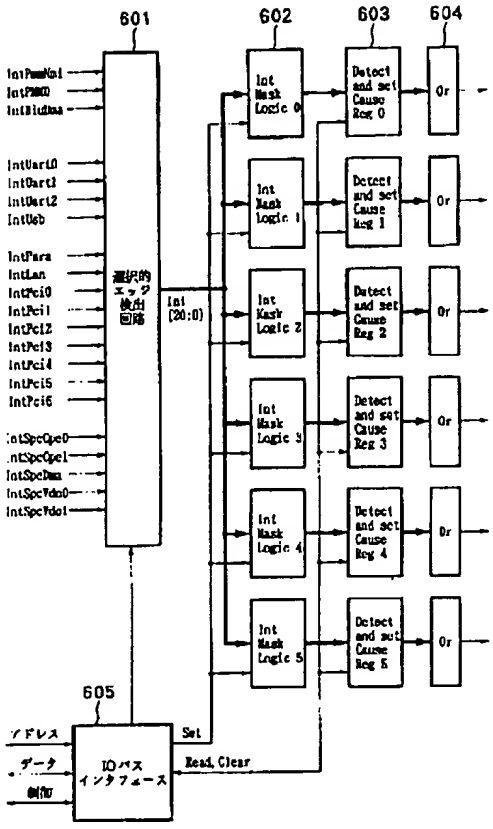
【図4】


SDRAM & ROM
CPU MIPS
401
403
402
400
404
406
407
408
405
409
410
411
412
413
414
415
416
Pバス
MCバス
スキャナ/プリンタ
コントローラ (SPC)
プリンタへの
スキャナTTD
からのビデオI/F

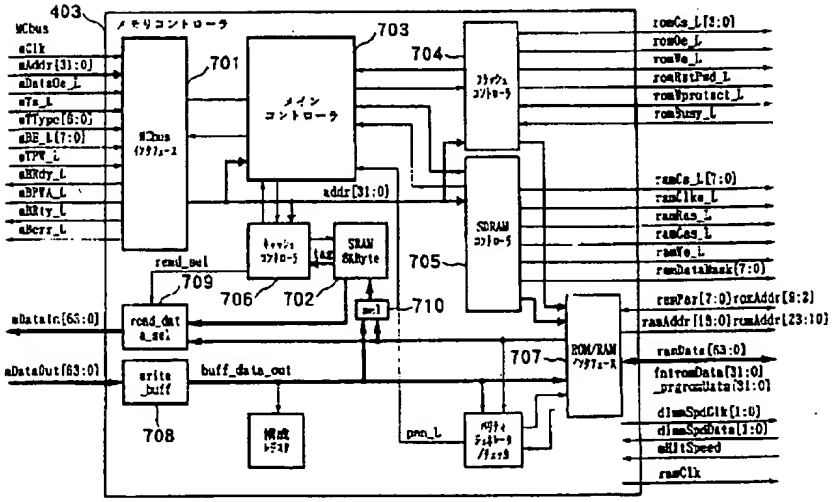
【図5】


Invalid
ロードミスの場合
キャッシュに読む
無効化
無効化
ストアヒット
Valid
Dirty
Valid
Clean
ロードミスの場合、
ライトバックして
キャッシュに読む
ロードミスの場合には
キャッシュに読む
ロードヒット
ストアミス
ロードヒット、
ストアミス、
ストアヒット

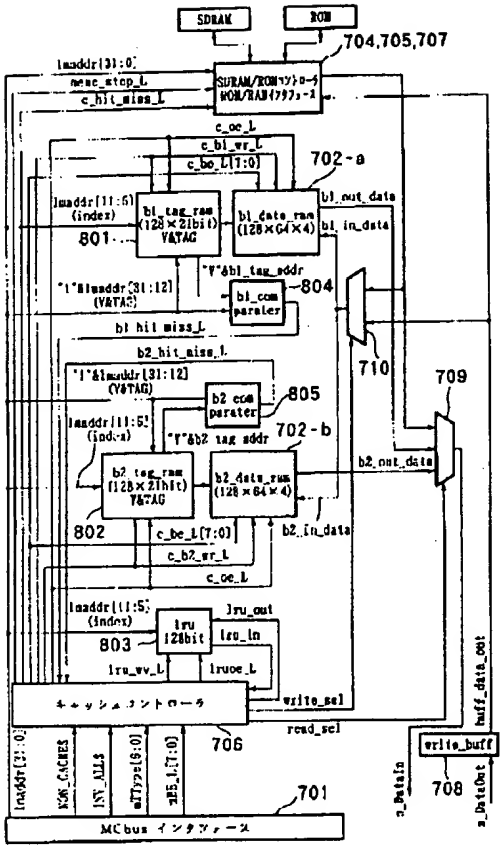
【図6】


601
602
603
604
605
選択的
エラー
検出
回路
Int
[20:0]
Int
Mask
Logic 0
Int
Mask
Logic 1
Int
Mask
Logic 2
Int
Mask
Logic 3
Int
Mask
Logic 4
Int
Mask
Logic 5
Detect
and set
Cause
Reg 0
Detect
and set
Cause
Reg 1
Detect
and set
Cause
Reg 2
Detect
and set
Cause
Reg 3
Detect
and set
Cause
Reg 4
Detect
and set
Cause
Reg 5
Or
IOバス
インタフェース
Set
Read Clear

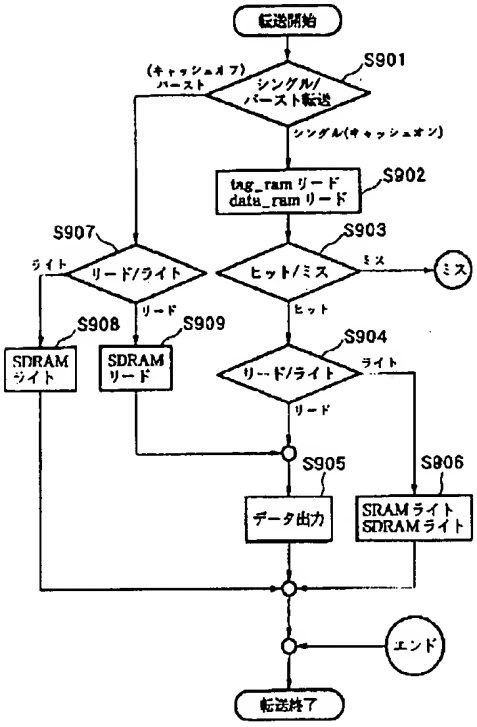
【図7】


403
メモリコントローラ
701
メイン
コントローラ
703
704
705
706
707
708
709
710
SDRAM
SRAM
addr[31:0]
read_out
write
buff_data_out
romCs_L[3:0]
romOe_L
romWe_L
romRstPwd_L
romWprotect_L
romBusy_L
ramCs_L[7:0]
ramClke_L
ramRas_L
ramCas_L
ramWe_L
ramDataMask[7:0]
ramBa[7:0]
ramAddr[12:0]
ramAddr[23:10]
ramData[63:0]
flromData[31:0]
_prgromData[31:0]
dimmSpdClk[1:0]
dimmSpdData[1:0]
mHzSpeed
ramClk

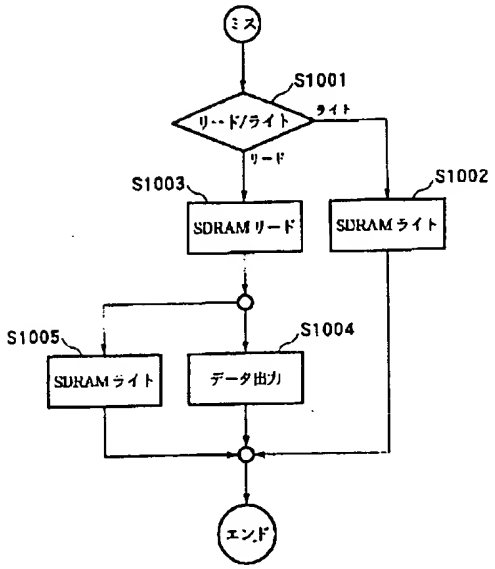
【図8】


SDRAM
ROM
704,705,707
SDRAM/ROMコントローラ
ROM/RAMインターフェース
c_oe_L
c_b1_wr_L
c_be_L[1:0]
702-a
b1_tag_ram (128×21bit) V&TAG
b1_data_ram (128×64×4)
b1_out_data
b1_in_data
801
b1_com parater
804
b1 hit miss_L
710
709
b2_hit_miss_L
b2 com parater
805
702-b
b2_tag_ram (128×21bit) V&TAG
b2_data_ram (128×64×4)
b2_out_data
b2_in_data
802
c_be_L[7:0]
c_b2_wr_L
c_oe_L
lru 128bit
803
lru_out
lru_in
lruwr_L
lruoe_L
キャッシュコントローラ
write_sel
read_sel
706
write_buff
buff_data_out
708
s_DataIn
s_DataOut
701
MCbus インタフェース

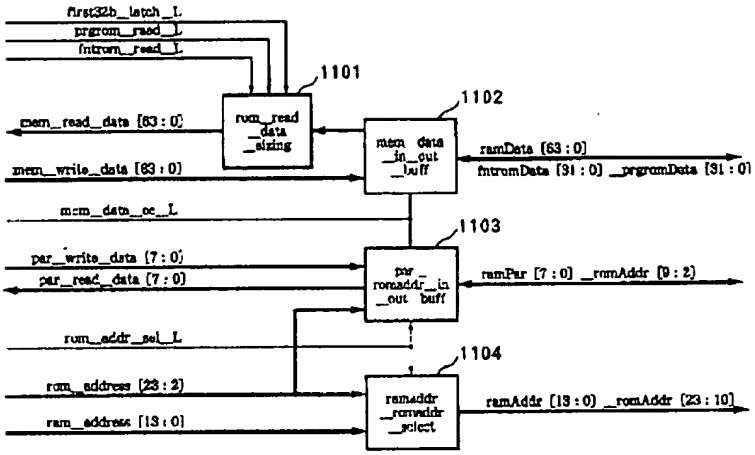
【図9】


転送開始
S901
(キャッシュオフ) バースト
シングル/バースト転送
シングル(キャッシュオン)
tag_ram リード data_ram リード
S902
S903
S907
ライト
リード/ライト
ヒット/ミス
ミス
ミス
リード
ヒット
S908
S909
S904
SDRAM ライト
SDRAM リード
リード/ライト
ライト
リード
S905
S906
データ出力
SRAMライト SDRAMライト
エンド
転送終了

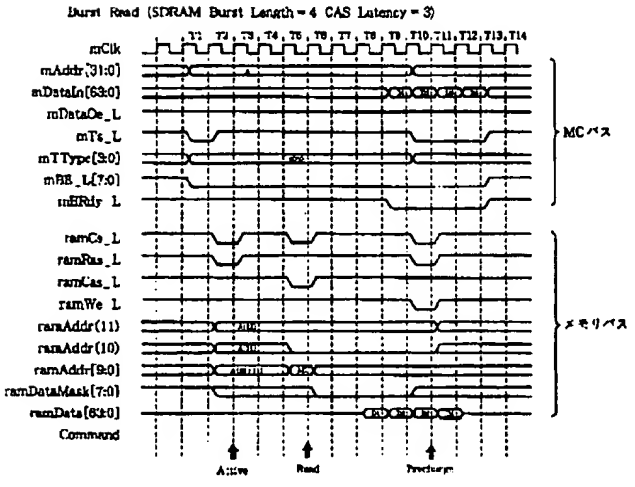
【図10】


ミス
S1001
リード/ライト
ライト
リード
S1003
S1002
SDRAM リード
SDRAM ライト
S1005
S1004
SDRAM ライト
データ出力
エンド

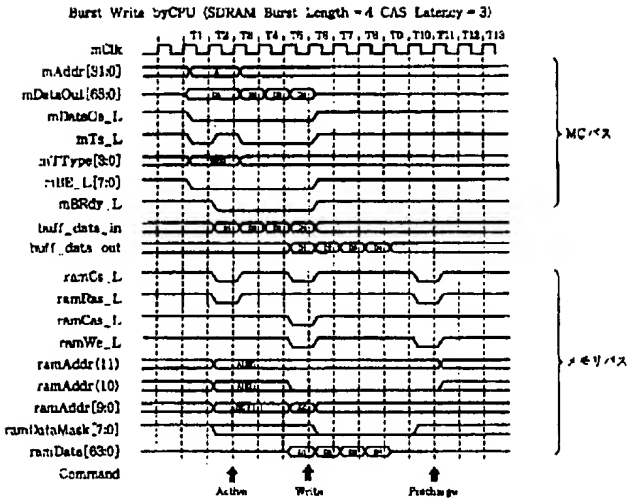
【図11】


first32b_latch_L
prgrom_read_L
fntrom_read_L
1101
rom_read_data_sizing
mem_read_data [63:0]
mem_write_data [63:0]
1102
mem data_in_out_buff
ramData [63:0]
fntromData [31:0] _prgromData [31:0]
mem_data_oe_L
1103
par_romaddr_in_out buff
par_write_data [7:0]
par_read_data [7:0]
ramPar [7:0] _romAddr [9:2]
rom_addr_sel_L
1104
romaddr_address [23:2]
ram_address [13:0]
ramaddr_romaddr_select
ramAddr [13:0] _romAddr [23:10]

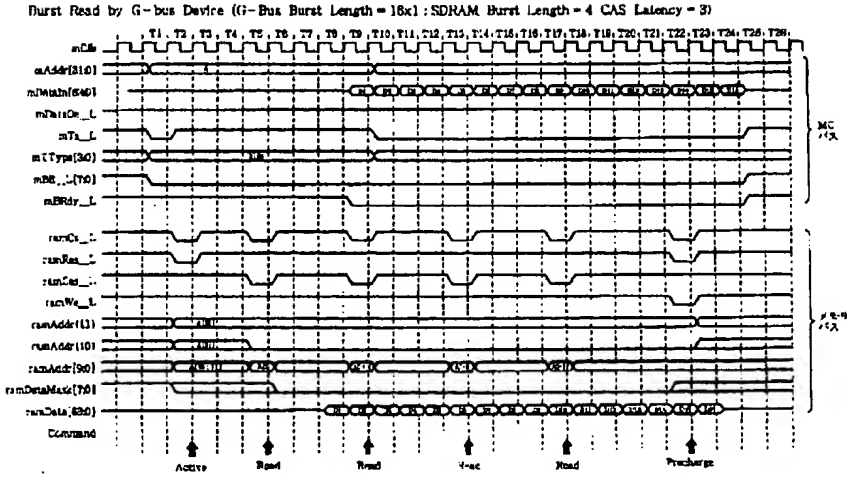
【図12】


Burst Read (SDRAM Burst Length = 4 CAS Latency = 3)
mClk
mAddr[31:0]
mDataIn[63:0]
mDataOe_L
mTs_L
mTType[3:0]
mBE_L[7:0]
mERdy_L
MCバス
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr(11)
ramAddr(10)
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
メモリバス
Command
Active
Read
Precharge

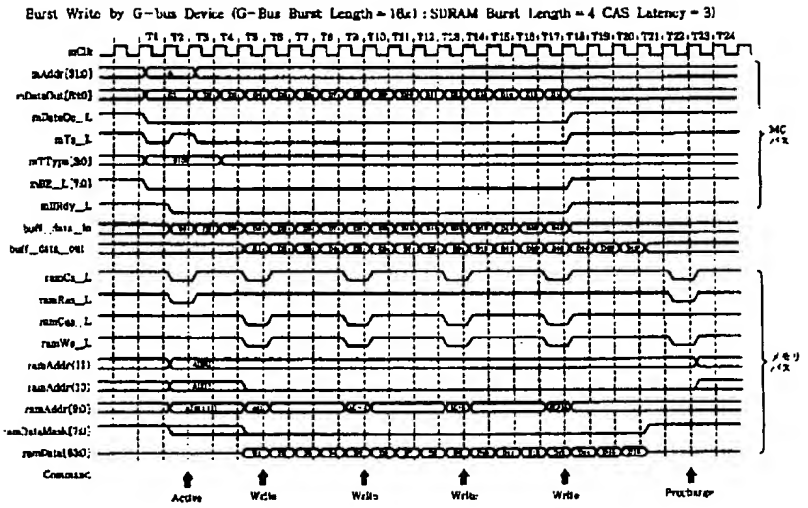
【図13】


Burst Write byCPU (SDRAM Burst Length = 4 CAS Latency = 3)
mClk
mAddr[31:0]
mDataOut[63:0]
mDataOe_L
mTs_L
mTType[3:0]
mBE_L[7:0]
mBRdy_L
buff_data_in
buff_data_out
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
Active
Write
Precharge
MCバス
メモリバス

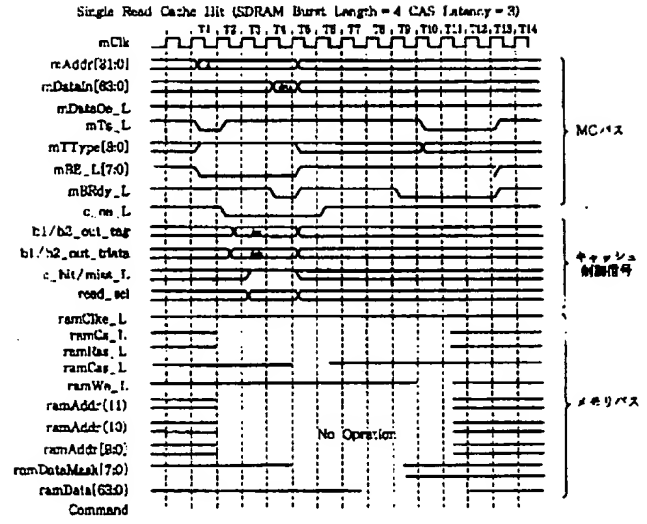
【図14】


Burst Read by G-bus Devire (G-Bus Burst Length = 16x1 : SDRAM Burst Length = 4 CAS Latency = 3)
mClk
mAddr[31:0]
mDataOe_L
mTs_L
mTType[3:0]
mBRdy_L
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
Active
Read
Read
Read
Precharge
MC
バス
メモリ
バス

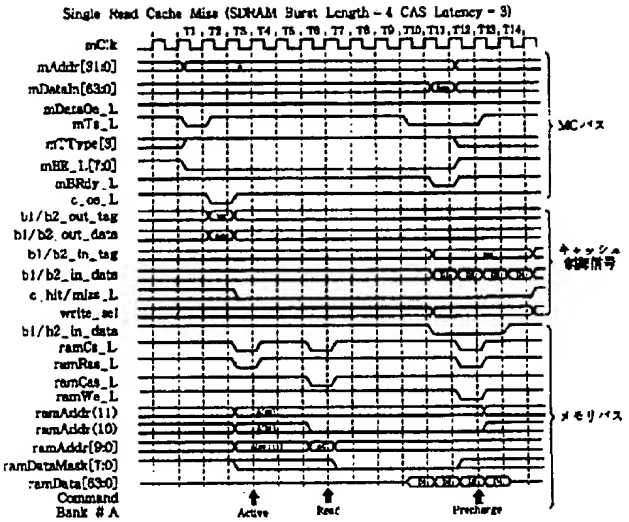
【図15】


Burst Write by G-bus Device (G-Bus Burst Length = 16x1 : SDRAM Burst Length = 4 CAS Latency = 3)
mClk
mAddr[31:0]
mDataOe_L
mTs_L
mTType[3:0]
mBE_L[7:0]
mBRdy_L
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
MC バス
メモリ バス
Active
Write
Write
Write
Write
Precharge

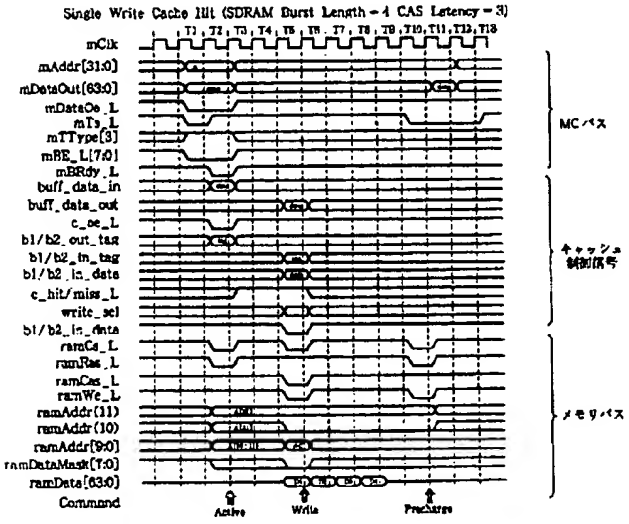
【図16】


Single Read Cache Hit (SDRAM Burst Length = 4 CAS Latency = 3)
mClk
mAddr[31:0]
mDataIn[63:0]
mDataOe_L
mTs_L
mTType[3:0]
mBE_L[7:0]
mBRdy_L
c_ns_L
b1/b2_out_tag
b1/b2_out_data
c_hit/miss_L
read_sel
ramClke_L
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
No Operation
MC バス
キャッシュ 制御信号
メモリバス

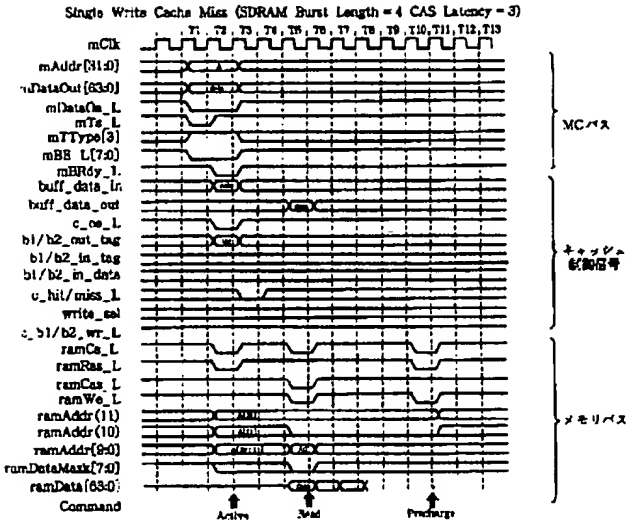
【図17】


Single Read Cache Miss (SDRAM Burst Length = 4 CAS Latency = 3)
T1 , T2 , T3 , T4 , T5 , T6 , T7 , T8 , T9 , T10, T11, T12, T13, T14,
mClk
mAddr[31:0]
mDataIn[63:0]
mDataOe_L
mTs_L
mTType[3]
mBE_L[7:0]
mBRdy_L
c_oe_L
b1/b2_out_tag
b1/b2_out_data
b1/b2_in_tag
b1/b2_in_data
c_hit/miss_L
write_sel
b1/b2_in_data
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
Bank # A
Active
Read
Precharge
MCバス
キャッシュ制御信号
メモリバス

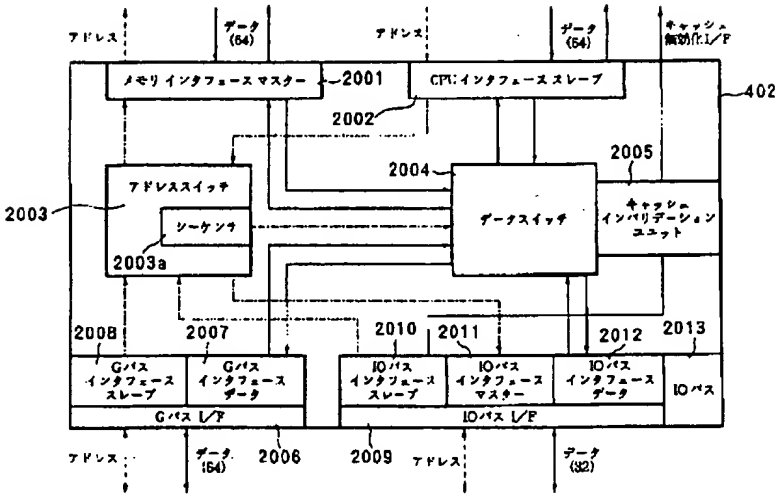
【図18】


Single Write Cache Hit (SDRAM Burst Length = 4 CAS Latency = 3)
T1 , T2 , T3 , T4 , T5 , T6 , T7 , T8 , T9 , T10, T11, T12, T13
mClk
mAddr[31:0]
mDataOut[63:0]
mDataOe_L
mTs_L
mTType[3]
mBE_L[7:0]
mBRdy_L
buff_data_in
buff_data_out
c_oe_L
b1/b2_out_tag
b1/b2_in_tag
b1/b2_in_data
c_hit/miss_L
write_sel
b1/b2_in_data
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
Active
Write
Precharge
MCバス
キャッシュ制御信号
メモリバス

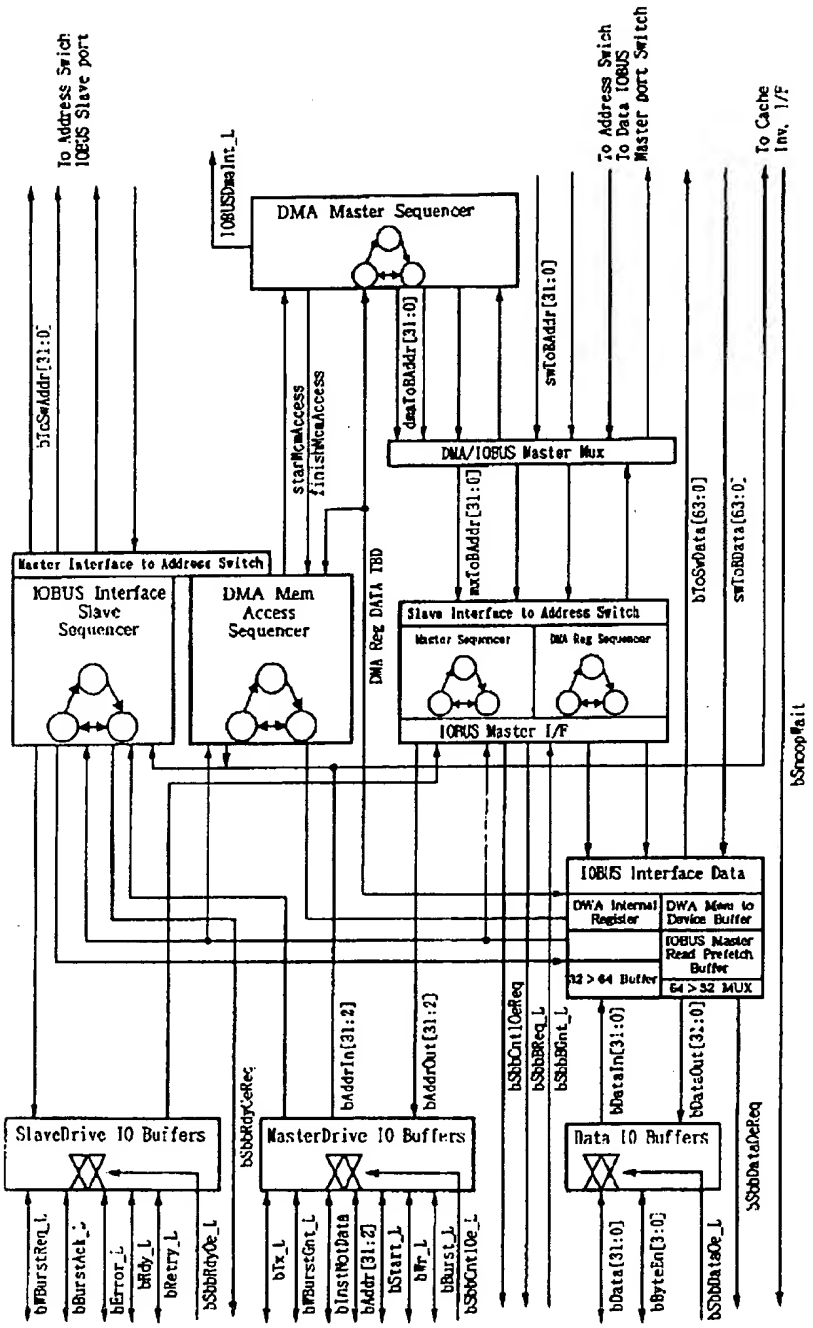
【図19】


Single Write Cache Miss (SDRAM Burst Length = 4 CAS Latency = 3)
T1 , T2 , T3 , T4 , T5 , T6 , T7 , T8 , T9 , T10, T11, T12, T13
mClk
mAddr[31:0]
mDataOut[63:0]
mDataOe_L
mTs_L
mTType[3]
mBB_L[7:0]
mBRdy_L
buff_data_in
buff_data_out
c_oe_L
b1/b2_out_tag
b1/b2_in_tag
b1/b2_in_data
c_hit/miss_L
write_sel
c_b1/b2_wr_L
ramCs_L
ramRas_L
ramCas_L
ramWe_L
ramAddr[11]
ramAddr[10]
ramAddr[9:0]
ramDataMask[7:0]
ramData[63:0]
Command
Active
Read
Precharge
MCバス
キャッシュ制御信号
メモリバス

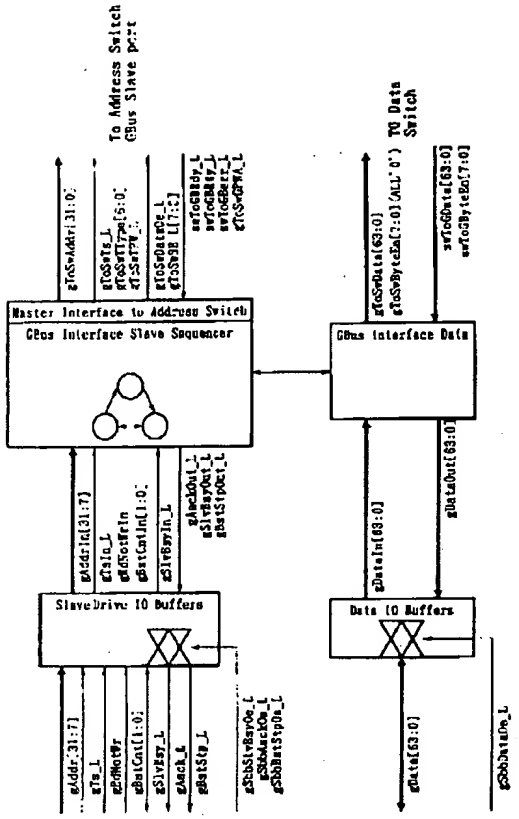
【図20】


アドレス
データ (64)
アドレス
データ (64)
キャッシュ制御I/F
メモリインタフェースマスター
2001
CPUインタフェーススレーブ
402
2002
2004
2005
アドレススイッチ
2003
シーケンサ
2003a
データスイッチ
キャッシュインバリデーションユニット
2008
2007
2010
2011
2012
2013
Gバス インタフェース スレーブ
Gバス インタフェース データ
IOバス インタフェース スレーブ
IOバス インタフェース マスター
IOバス インタフェース データ
IOバス
Gバス I/F
IOバス I/F
アドレス
データ (64)
2006
2009
アドレス
データ (32)

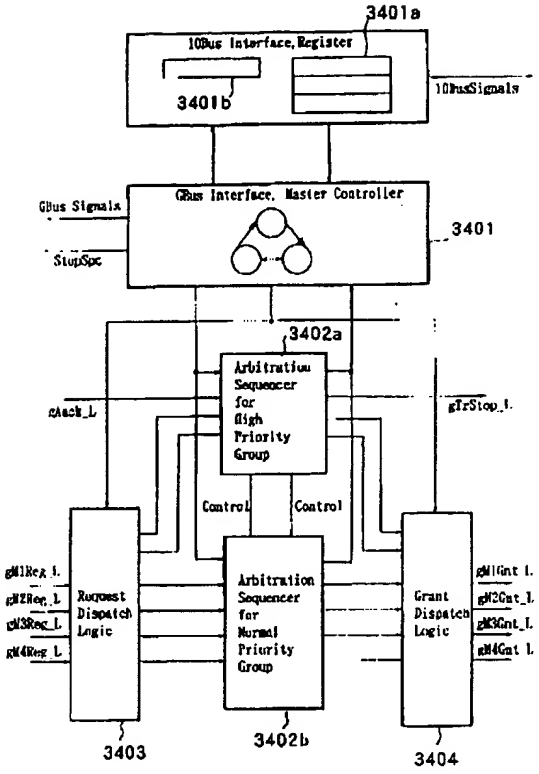
【図21】


To Address Switch
IOBUS Slave port
IOBUSDmaInt_L
DMA Master Sequencer
To Address Switch
To Data IOBUS
Master port Switch
To Cache
Inv, I/F
bToSwAddr[31:0]
startMemAccess
finishMemAccess
dmaToBAddr[31:0]
swToBAddr[31:0]
DMA/IOBUS Master Mux
bToSwData[63:0]
swToBData[63:0]
bSnoopWait
Master Interface to Address Switch
IOBUS Interface
Slave
Sequencer
DMA Mem
Access
Sequencer
DMA Reg DATA TBD
muxToBAddr[31:0]
Slave Interface to Address Switch
Master Sequencer
DMA Reg Sequencer
IOBUS Master I/F
IOBUS Interface Data
DMA Internal
Register
DMA Mem to
Device Buffer
IOBUS Master
Read Prefetch
Buffer
32 > 64 Buffer
64 > 32 MUX
bSbbCntlOeReq
bSbbBReq_L
bSbbBGnt_L
bDataIn[31:0]
bDataOut[31:0]
bSbbRdyOeReq
bAddrIn[31:2]
bAddrOut[31:2]
SlaveDrive IO Buffers
MasterDrive IO Buffers
Data IO Buffers
bSbbDataOeReq
bFBurstReq_L
bBurstAck_L
bError_L
bRdy_L
bRetry_L
bSbbRdyOe_L
bTx_L
bFBurstGnt_L
bInstNotData
bAddr[31:2]
bStart_L
bRr_L
bBurst_L
bSbbCntlOe_L
bData[31:0]
bByteEn[3:0]
bSbbDataOe_L

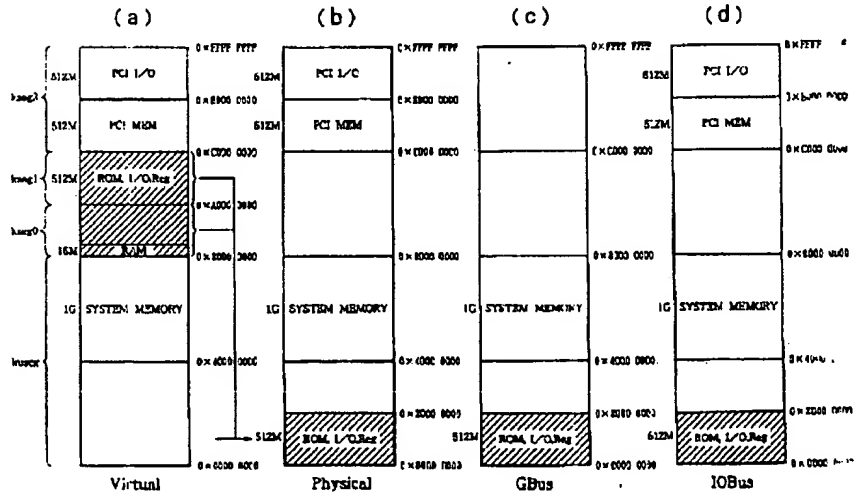
【図22】


To Address Switch GBus Slave port
TO Data Switch
Master Interface to Address Switch
GBus Interface Slave Sequencer
GBus Interface Data
SlaveDrive IO Buffers
Data IO Buffers

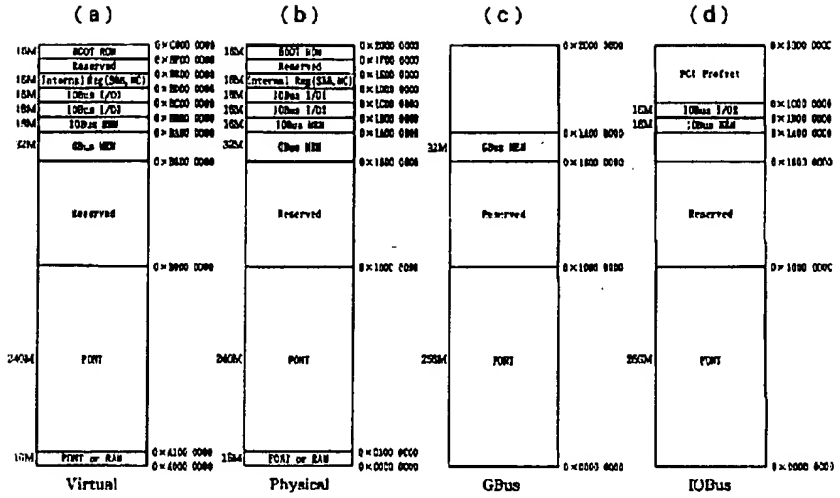
【図34】


3401a
IOBus Interface, Register
3401b
IOBusSignals
GBus Signals
GBus Interface, Master Controller
3401
StopSpc
3402a
Arbitration Sequencer for High Priority Group
gAack_L
gTrStop_L
Control
Control
Request Dispatch Logic
Arbitration Sequencer for Normal Priority Group
Grant Dispatch Logic
3403
3402b
3404

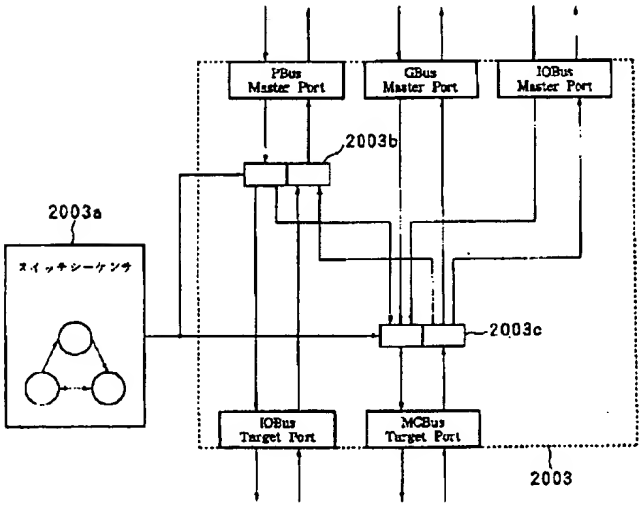
【図23】


(a)
(b)
(c)
(d)
PCI I/O
PCI MEM
ROM, I/ORes
RAM
SYSTEM MEMORY
Virtual
Physical
GBus
IOBus

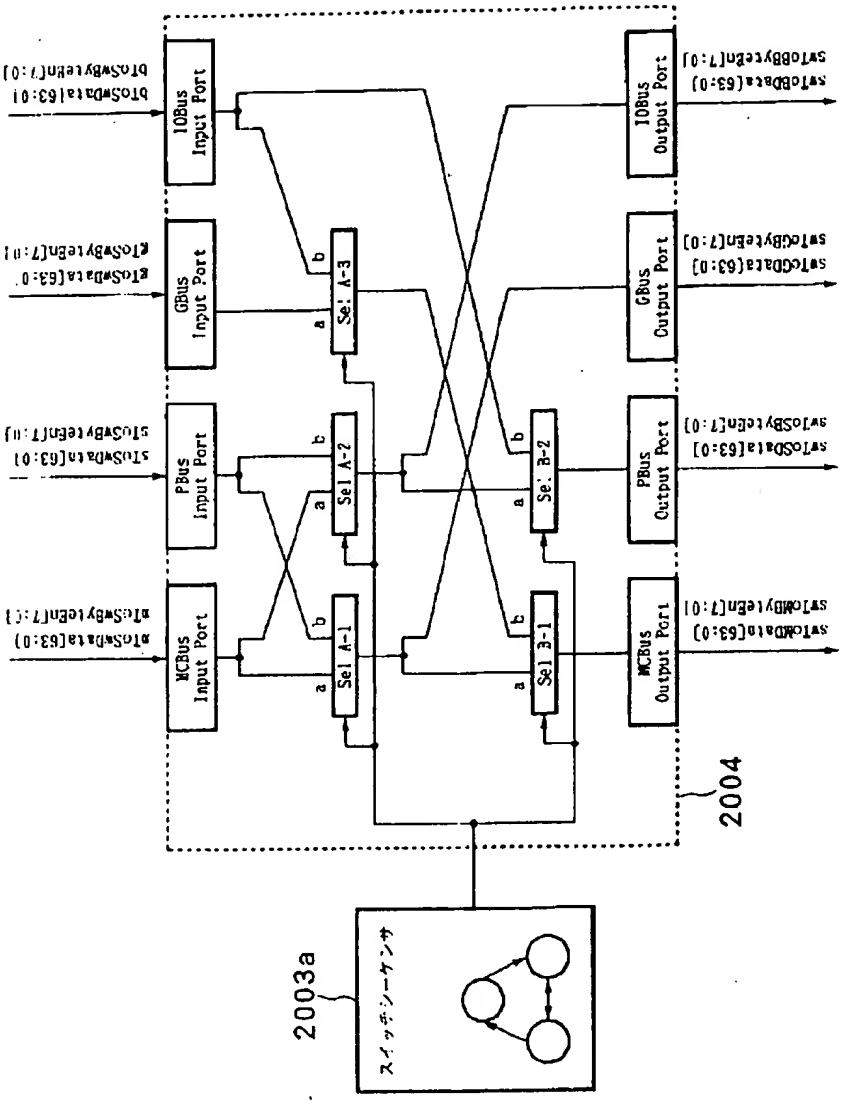
【図24】


(a)
(b)
(c)
(d)
Virtual
Physical
GBus
IOBus

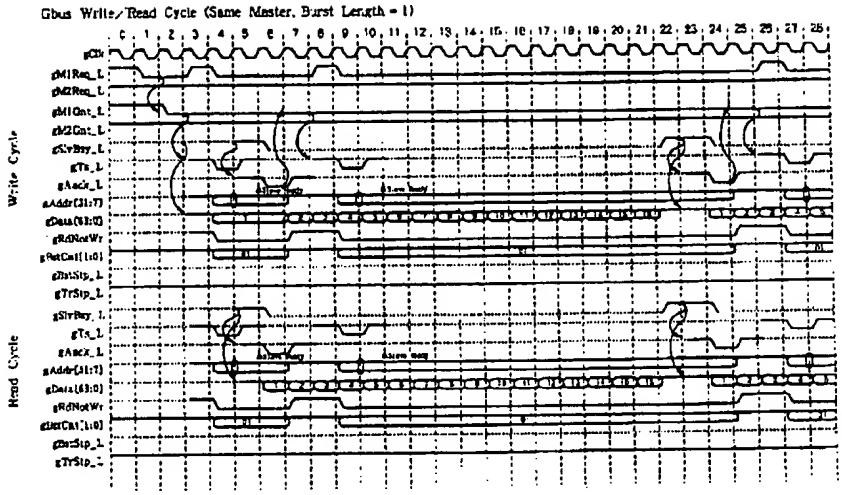
【図25】


PBus Master Port
GBus Master Port
IOBus Master Port
2003b
2003a
スイッチシーケンサ
2003c
IOBus Target Port
MCBus Target Port
2003

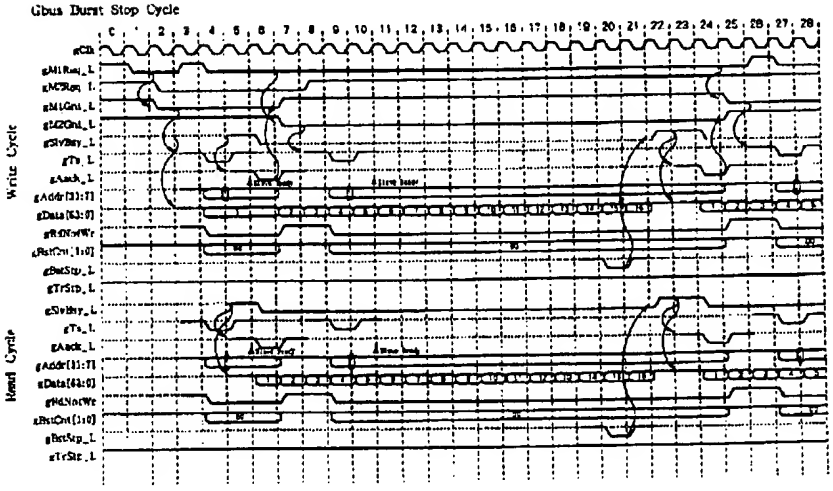
【図26】


10Bus Input Port
GBus Input Port
PBus Input Port
MCBus Input Port
10Bus Output Port
GBus Output Port
PBus Output Port
MCBus Output Port
Sel A-3
Sel A-2
Sel A-1
Sel B-2
Sel B-1
2004
2003a
スイッチシーケンサ

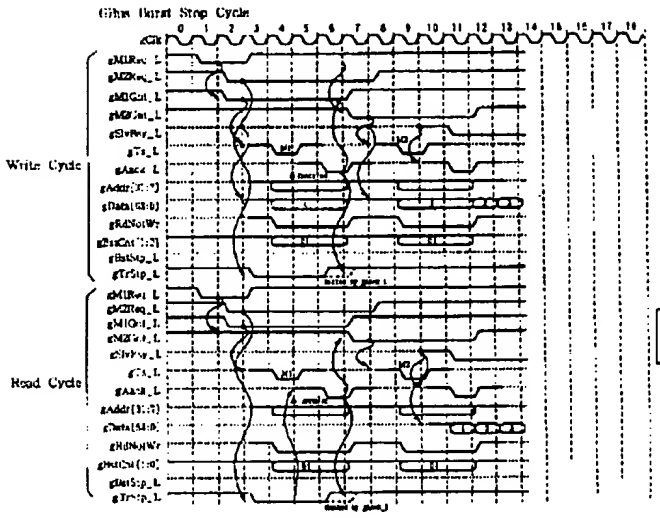
【図27】


Gbus Write/Read Cycle (Same Master, Burst Length = 1)
Write Cycle
Read Cycle

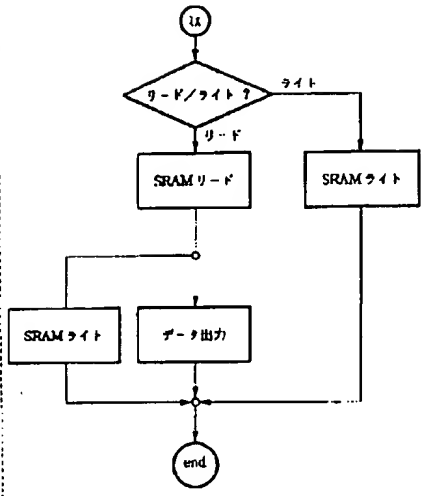
【図28】


Gbus Burst Stop Cycle
Write Cycle
Read Cycle

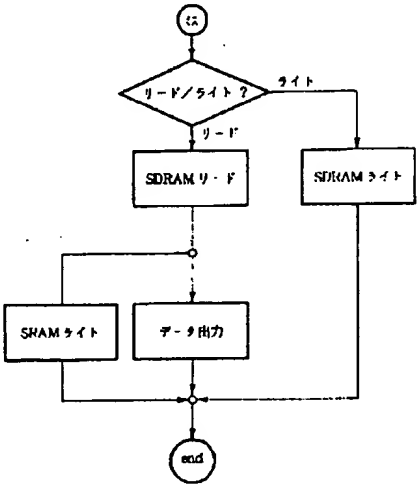
【図29】



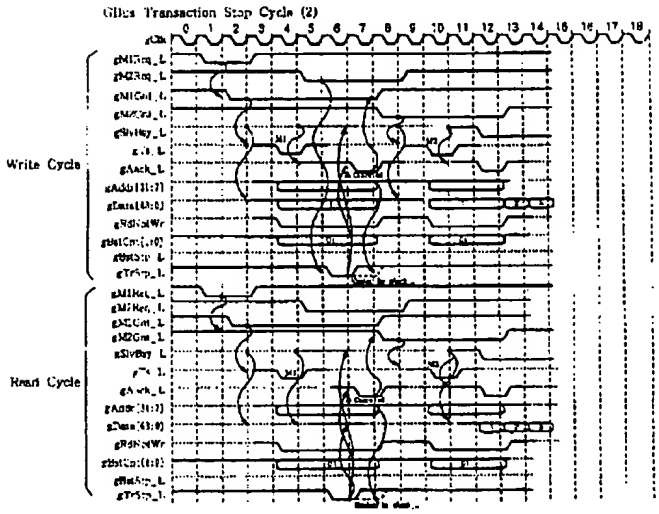
【図57】



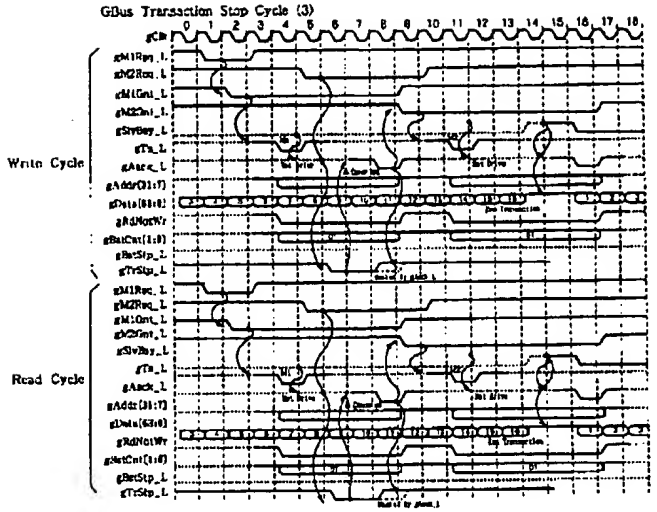
【図59】



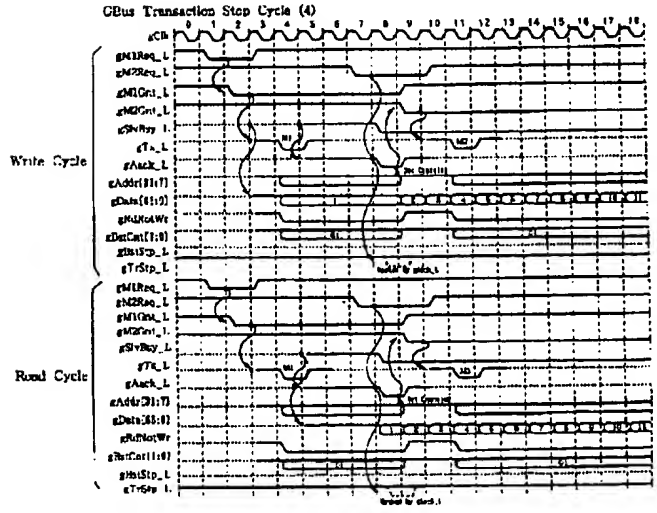
【図30】



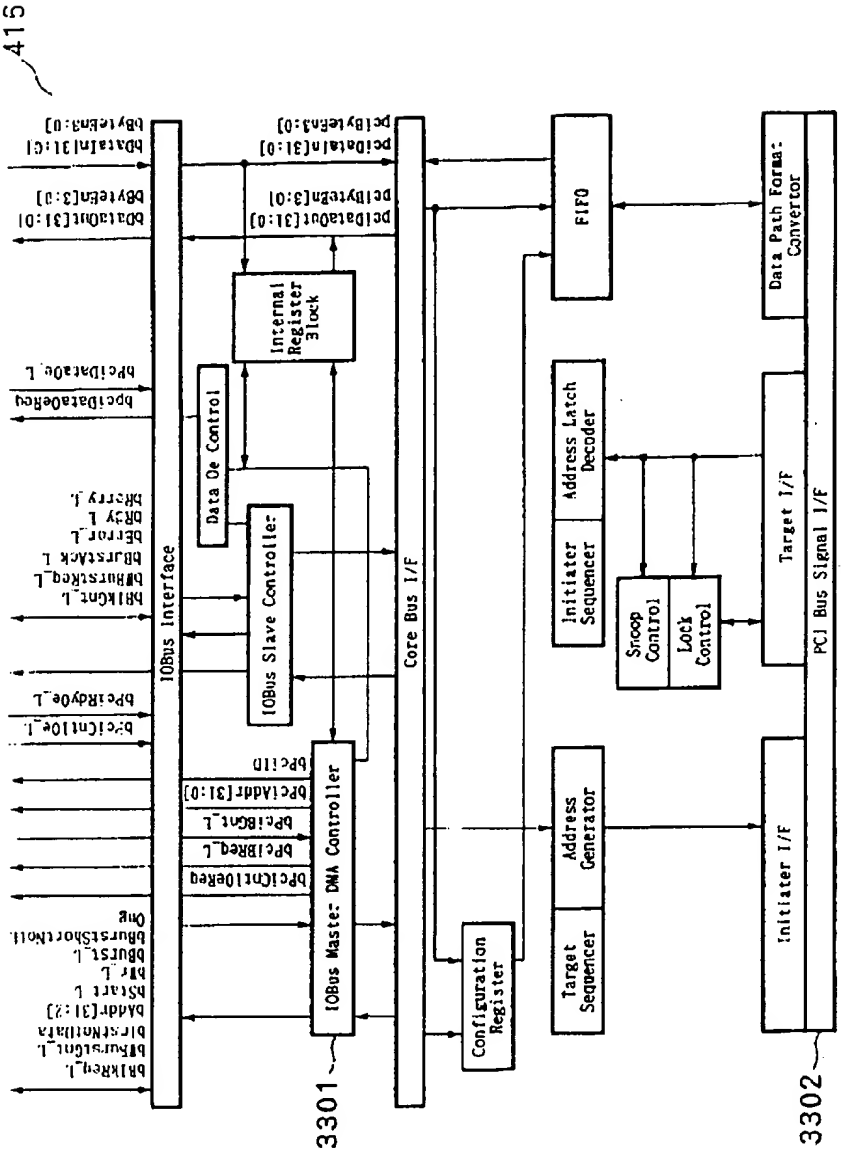
【図31】


GBus Transaction Step Cycle (3)
Write Cycle
Read Cycle

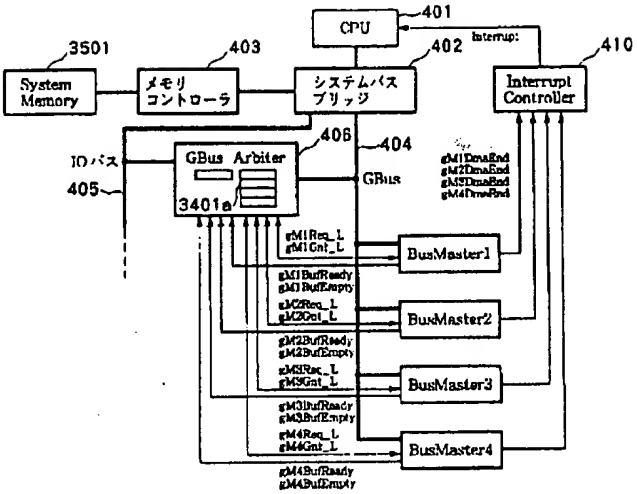
【図32】


GBus Transaction Step Cycle (4)
Write Cycle
Read Cycle

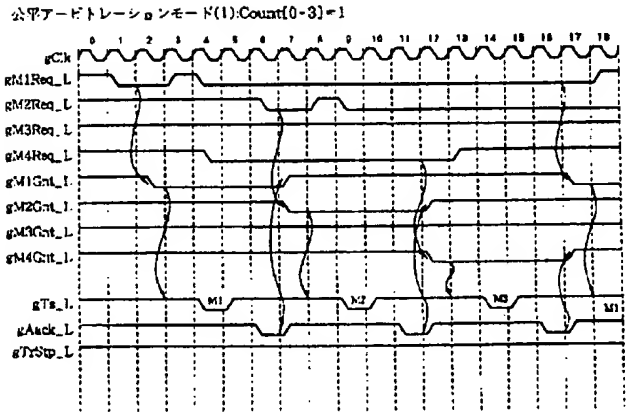
【図33】


415
IOBus Interface
Core Bus I/F
Internal Register Block
Data Oe Control
IOBus Slave Controller
IOBus Master DMA Controller
Configuration Register
FIFO
Address Latch Decoder
Initiater Sequencer
Snoop Control
Lock Control
Address Generator
Target Sequencer
Target I/F
Initiater I/F
Data Path Format Convertor
PCI Bus Signal I/F
3301
3302

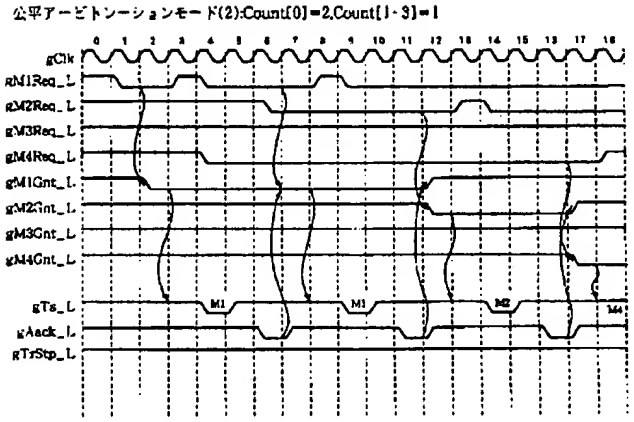
【図３５】


CPU
401
Interrupt
3501
403
402
410
System Memory
メモリコントローラ
システムバスブリッジ
Interrupt Controller
406
404
IOバス
GBus Arbiter
GBus
405
3401a
BusMaster1
BusMaster2
BusMaster3
BusMaster4

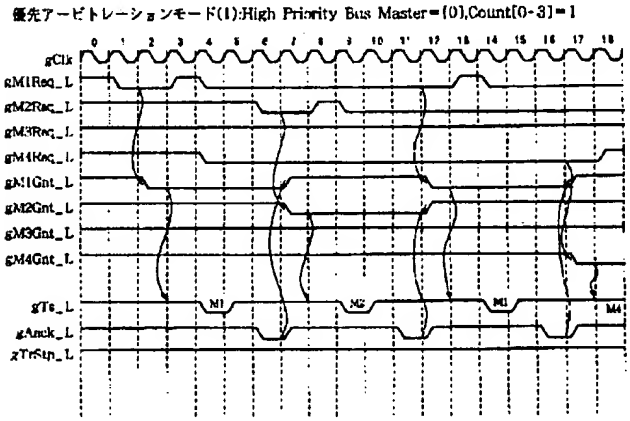
【図３６】


gClk
gM1Req_L
gM2Req_L
gM3Req_L
gM4Req_L
gM1Gnt_L
gM2Gnt_L
gM3Gnt_L
gM4Gnt_L
gTs_L
gAack_L
gTrStp_L

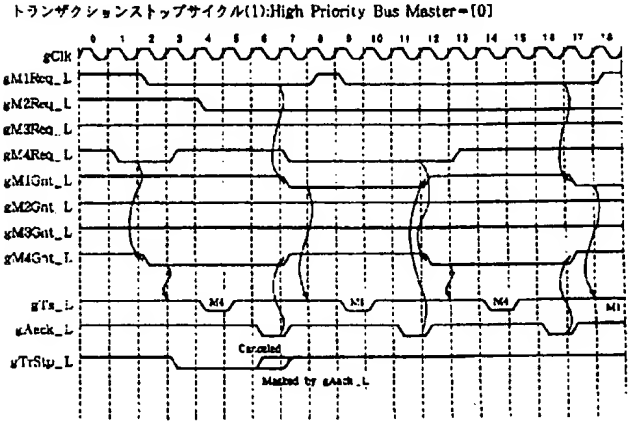
【図37】


公平アービトレーションモード(2):Count[0]=2,Count[1-3]=1
gClk
gM1Req_L
gM2Req_L
gM3Req_L
gM4Req_L
gM1Gnt_L
gM2Gnt_L
gM3Gnt_L
gM4Gnt_L
gTs_L
gAack_L
gTrStp_L

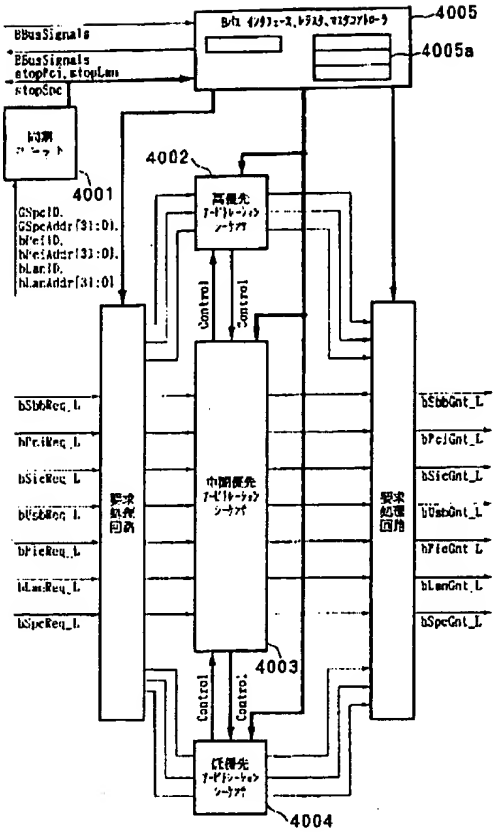
【図38】


優先アービトレーションモード(1):High Priority Bus Master=[0],Count[0-3]=1
gClk
gM1Req_L
gM2Req_L
gM3Req_L
gM4Req_L
gM1Gnt_L
gM2Gnt_L
gM3Gnt_L
gM4Gnt_L
gTs_L
gAack_L
gTrStp_L

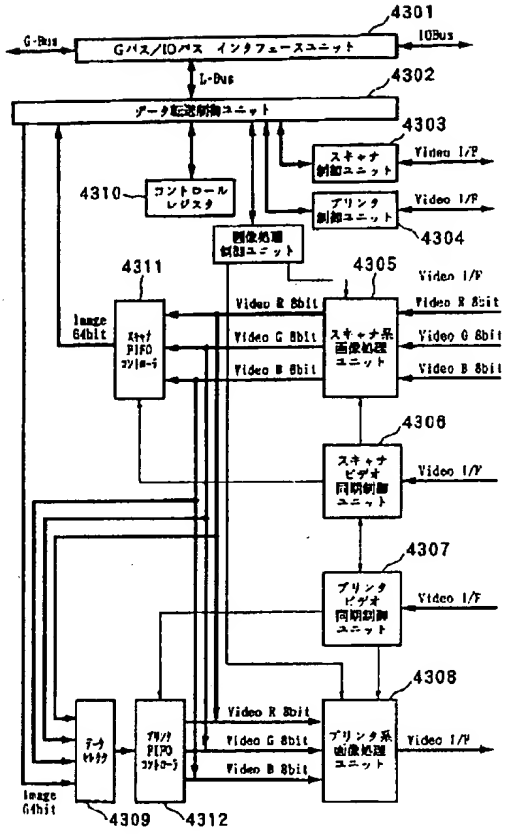
【図39】


トランザクションストップサイクル(1):High Priority Bus Master=[0]
gClk
gM1Req_L
gM2Req_L
gM3Req_L
gM4Req_L
gM1Gnt_L
gM2Gnt_L
gM3Gnt_L
gM4Gnt_L
gTs_L
gAack_L
gTrStp_L
Canceled
Masked by gAack_L

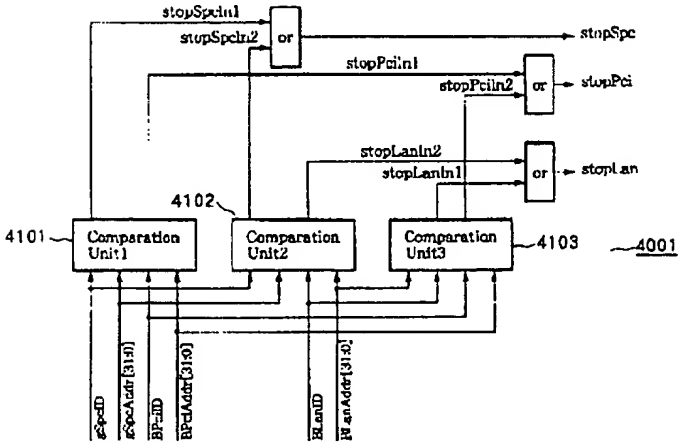
【図40】


4005
4005a
4001
4002
4003
4004
Control
bSbbReq_L
bPciReq_L
bSicReq_L
bUsbReq_L
bPieReq_L
bLanReq_L
bSpcReq_L
bSbbGnt_L
bPciGnt_L
bSicGnt_L
bUsbGnt_L
bPieGnt_L
bLanGnt_L
bSpcGnt_L

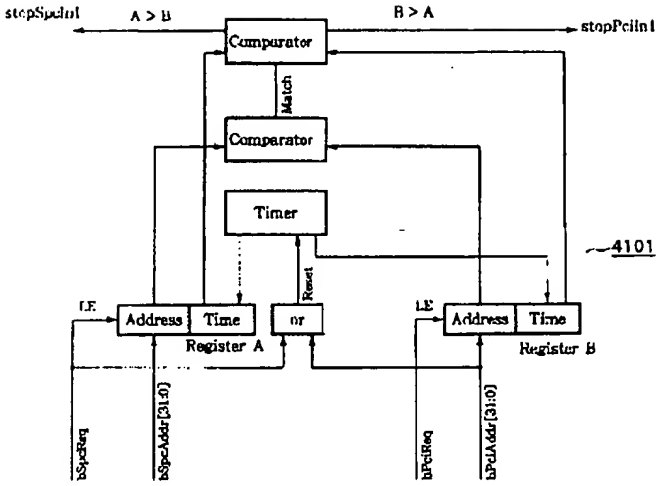
【図43】


4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
G-Bus
I-Bus
L-Bus
Video I/F
Video R 8bit
Video G 8bit
Video B 8bit
Image 64bit

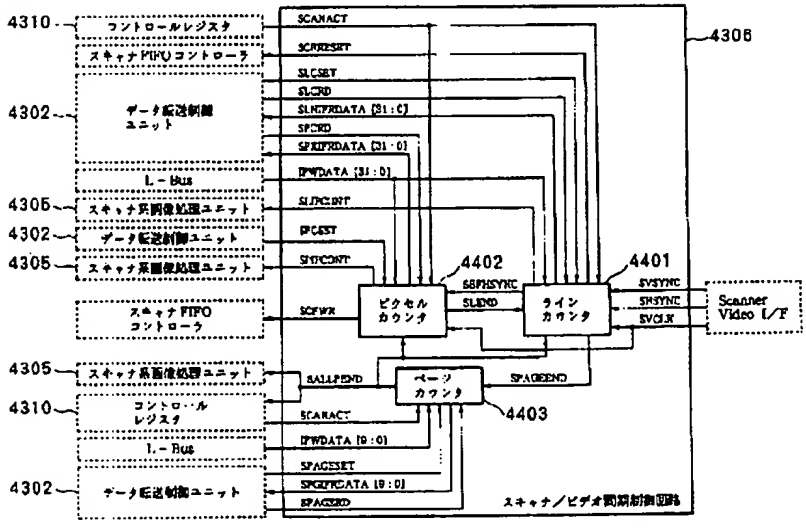
【図41】


stopSpcIn1
stopSpcIn2
or
stopSpc
stopPciIn1
stopPciIn2
stopPci
stopLanIn2
stopLanIn1
stopLan
4102
4101
Comparation Unit1
Comparation Unit2
Comparation Unit3
4103
4001

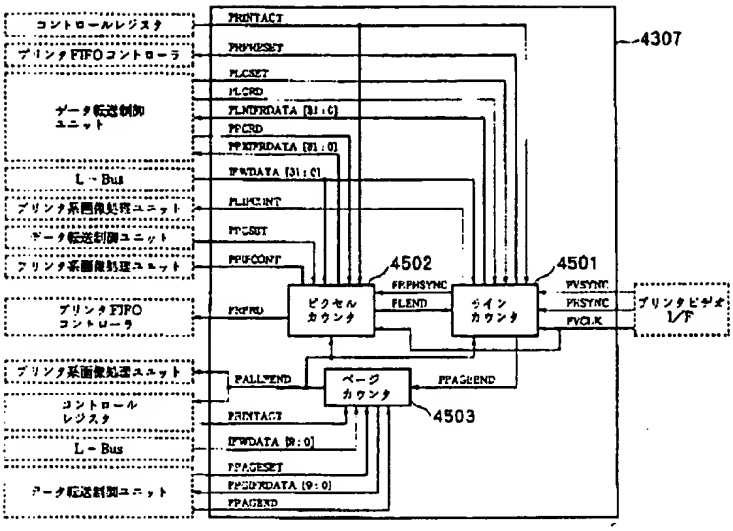
【図42】


stopSpdInt
A > B
B > A
stopPciInt
Comparator
Match
Comparator
Timer
Reset
4101
LE
Address
Time
Register A
or
LE
Address
Time
Register B
bSpcReq
bSpcAddr[31:0]
bPciReq
bPciAddr[31:0]

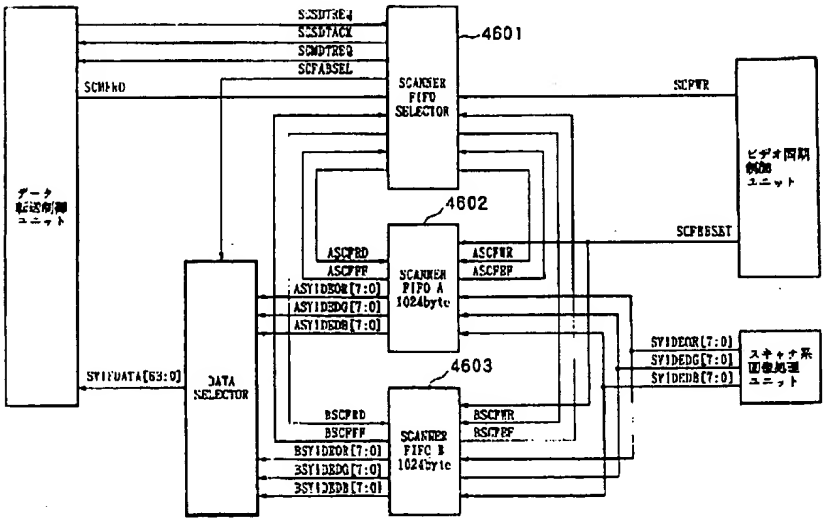
【図44】


4310
コントロールレジスタ
スキャナFIFOコントローラ
4302
データ転送制御ユニット
L-Bus
4305
スキャナ画像処理ユニット
4302
データ転送制御ユニット
4305
スキャナ画像処理ユニット
スキャナFIFOコントローラ
4305
スキャナ画像処理ユニット
4310
コントロールレジスタ
L-Bus
4302
データ転送制御ユニット
SCANACT
SCRRESET
SLCSET
SLCRD
SLRDFRDATA [31:0]
SFCRD
SFRDFRDATA [31:0]
IFWDATA [31:0]
SLFCENT
IFCSET
SYNCONT
SCFWR
SALLPEND
SCANACT
IFWDATA [9:0]
SPAGESET
SPGSYNCDATA [9:0]
SPAGERD
4402
ピクセルカウンタ
SSPHSYNC
SLEND
4401
ラインカウンタ
SPAGEEND
4403
ページカウンタ
4306
SVSYNC
SHSYNC
SVCLK
Scanner Video I/F
スキャナ/ビデオ同期制御回路

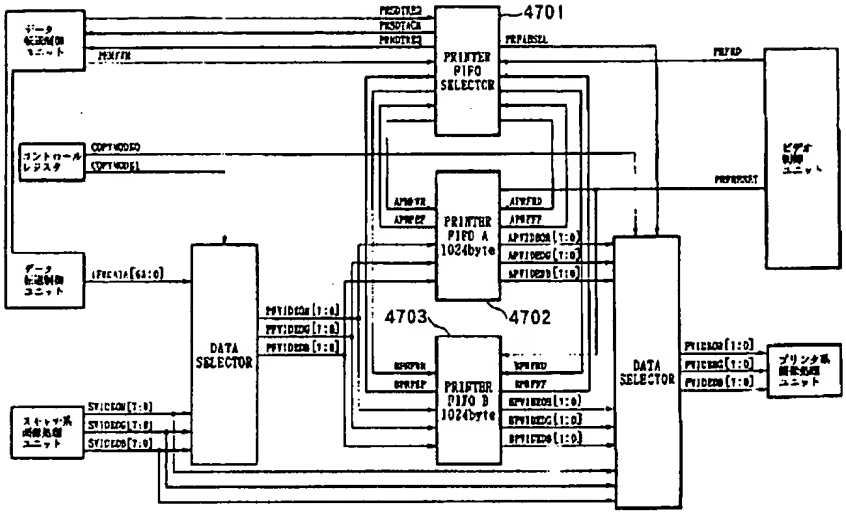
【図45】


4307
4502
4501
4503
ピクセル
カウンタ
ライン
カウンタ
ページ
カウンタ
プリンタビデオ
I/F
L-Bus

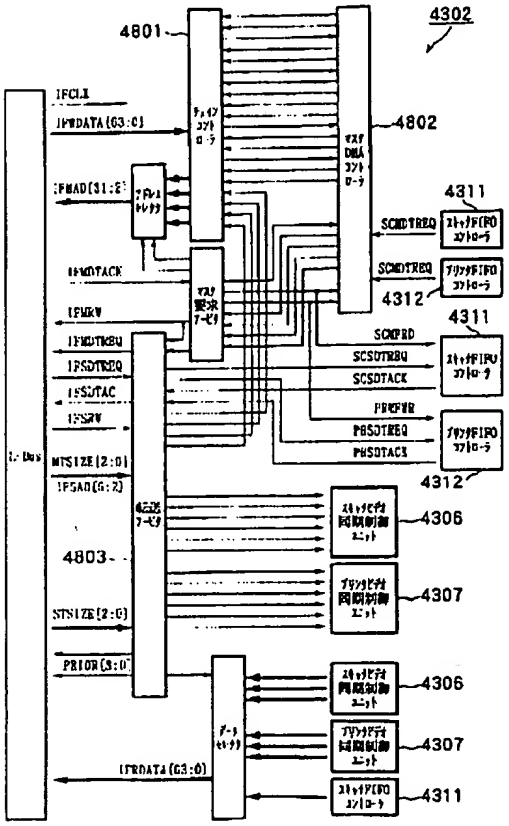
【図46】


4601
4602
4603
SCANNER
FIFO
SELECTOR
SCANNER
FIFO A
1024byte
SCANNER
FIFO B
1024byte
DATA
SELECTOR
データ
転送制御
ユニット
ビデオ同期
制御
ユニット
スキャナ系
画像処理
ユニット
L-Bus

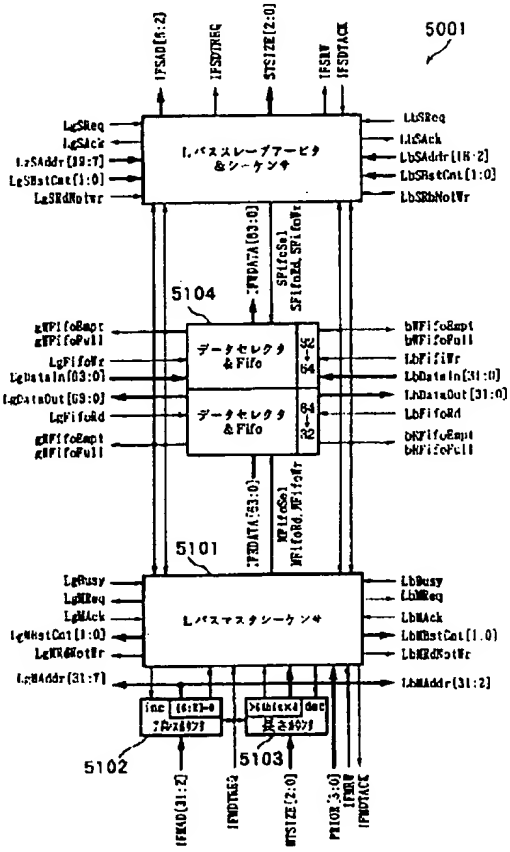
【図47】


4701
PRINTER FIFO SELECTOR
PRINTER FIFO A 1024byte
PRINTER FIFO B 1024byte
4702
4703
DATA SELECTOR
DATA SELECTOR

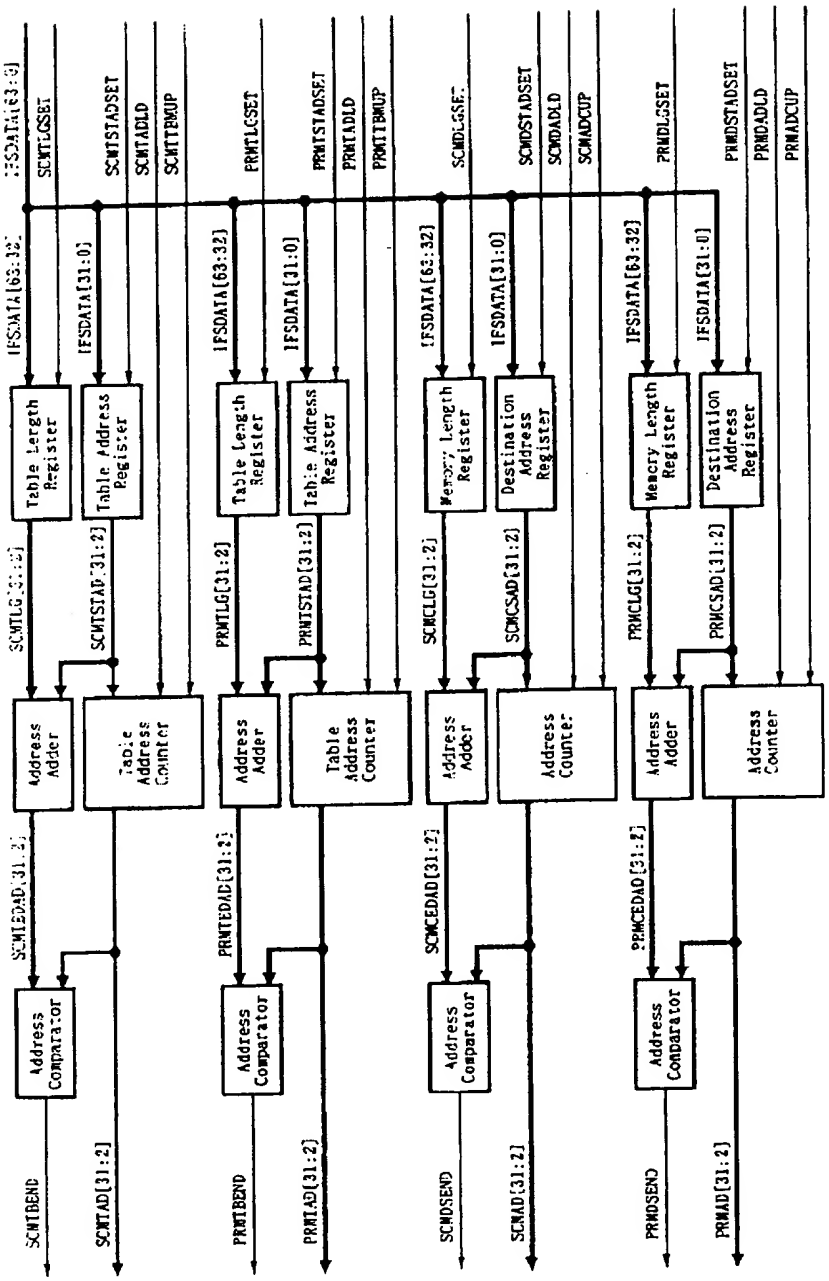
【図48】


4801
4802
4302
4311
4312
4306
4307
4803
IFCLK
IFWDATA[63:0]
IFRDATA[63:0]
IFMRV
IFSDTREQ
IFSDTAC
IFSRV
MTSIZE[2:0]
IFSAD[6:2]
STSIZE[2:0]
PRIOR[3:0]

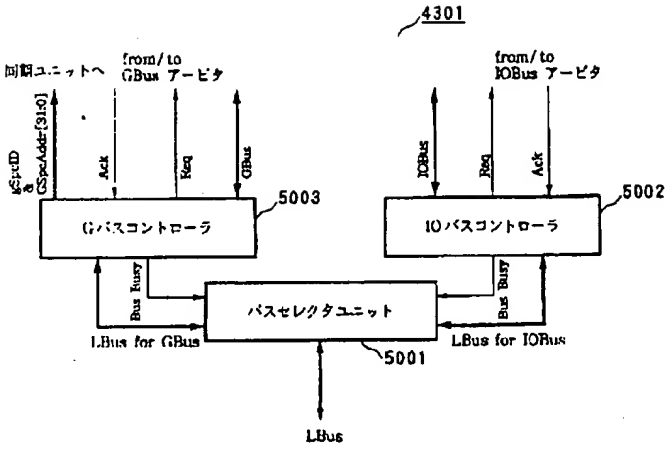
【図51】


5001
5101
5102
5103
5104
Lバススレーブアービタ &シーケンサ
Lバスマスタシーケンサ
データセレクタ &Fifo
データセレクタ &Fifo
IFSAD[6:2]
IFSDTREQ
STSIZE[2:0]
IFSRV
IFSDTACK
IFWDATA[63:0]
IFRDATA[63:0]
IFRAD[31:2]
PRIOR[3:0]

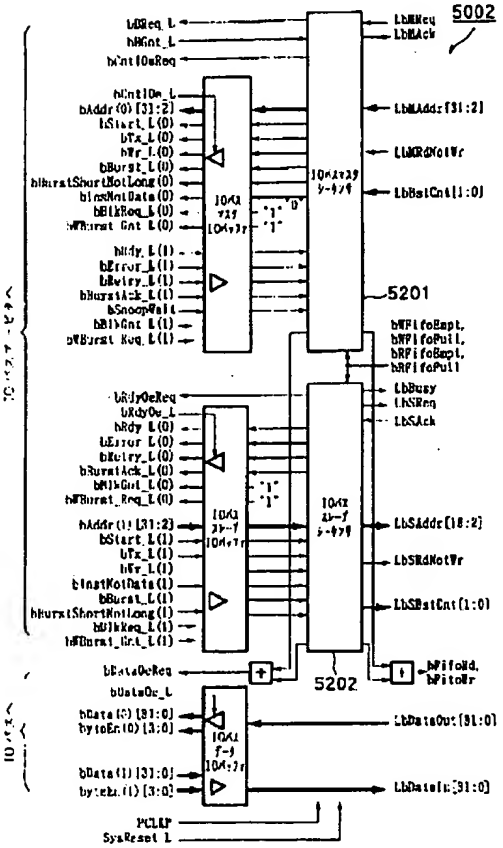
【図49】


IFSDATA[63:32]
SCMTLGSET
SCMTSTADSET
SCMTADLD
SCMTTBMUP
PRMTLGSET
PRMTSTADSET
PRMTADLD
PRMTTBMUP
SCMDLGSET
SCMDSTADSET
SCMDADLD
SCMADCUP
PRMDLGSET
PRMDSTADSET
PRMDADLD
PRMADCUP
IFSDATA[31:0]
Table Length Register
Table Address Register
Memory Length Register
Destination Address Register
SCMTLG[31:2]
SCMTSTAP[31:2]
PRMTLG[31:2]
PRMTSTAD[31:2]
SCMCLG[31:2]
SCMCSAD[31:2]
PRMCLG[31:2]
PRMCSAD[31:2]
Address Adder
Table Address Counter
Address Counter
SCMTEDAD[31:2]
PRMTEDAD[31:2]
SCMCEDAD[31:2]
PRMCEDAD[31:2]
Address Comparator
SCMTBEND
SCMTAD[31:2]
PRMTBEND
PRMTAD[31:2]
SCMDSEND
SCMAD[31:2]
PRMDSEND
PRMAD[31:2]

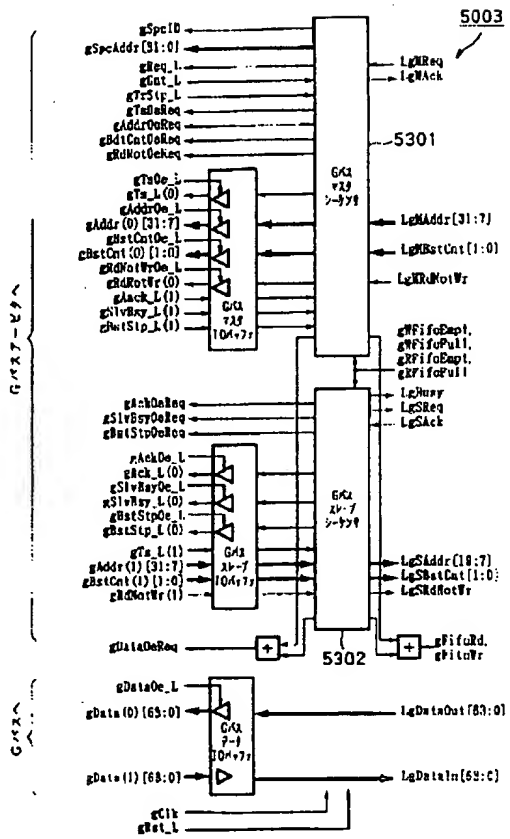
【図50】


4301
回路ユニットへ
from/to GBus アービタ
from/to IOBus アービタ
gSrcID & gSpecAddr[31:0]
Ack
Req
GBus
IOBus
5003
Gバスコントローラ
5002
IOバスコントローラ
Bus Busy
バスセレクタユニット
LBus for GBus
LBus for IOBus
5001
LBus

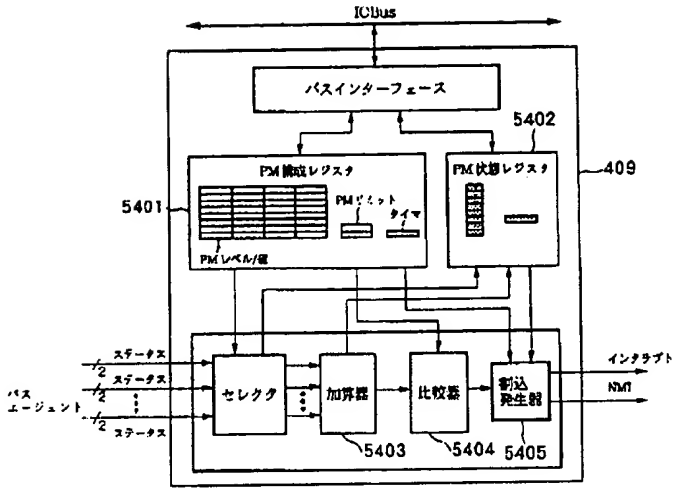
【図52】


5002
5201
5202
IOバスへ
PCLK
SysReset_L

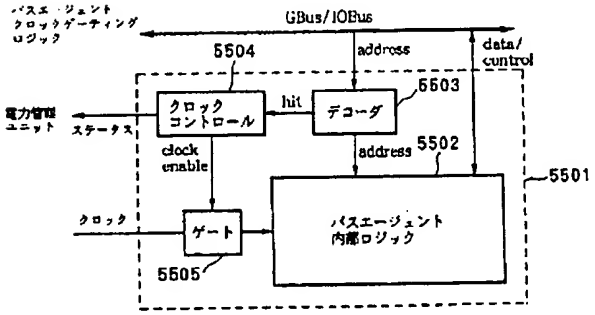
【図53】


5003
5301
5302
Gバスへ
gClk
gRst_L

【図54】

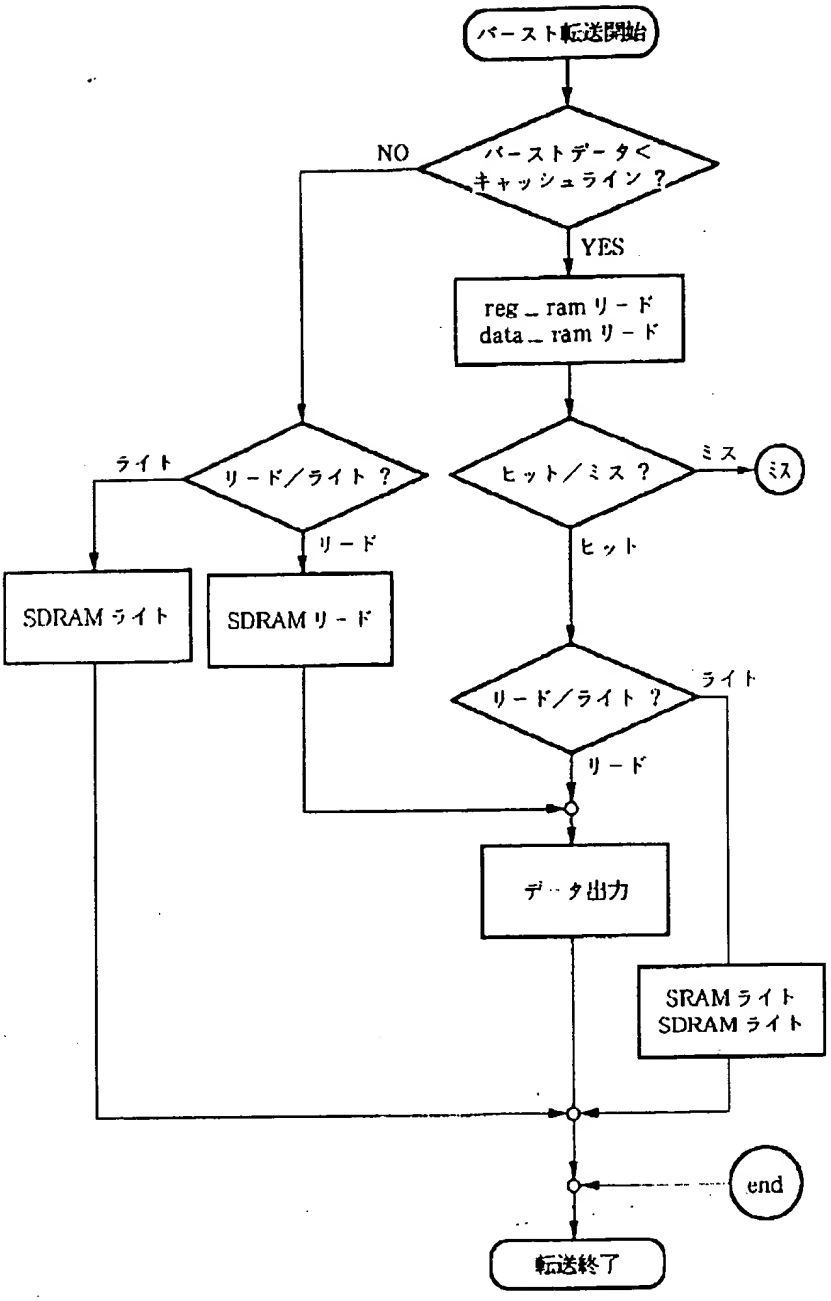


【図55】

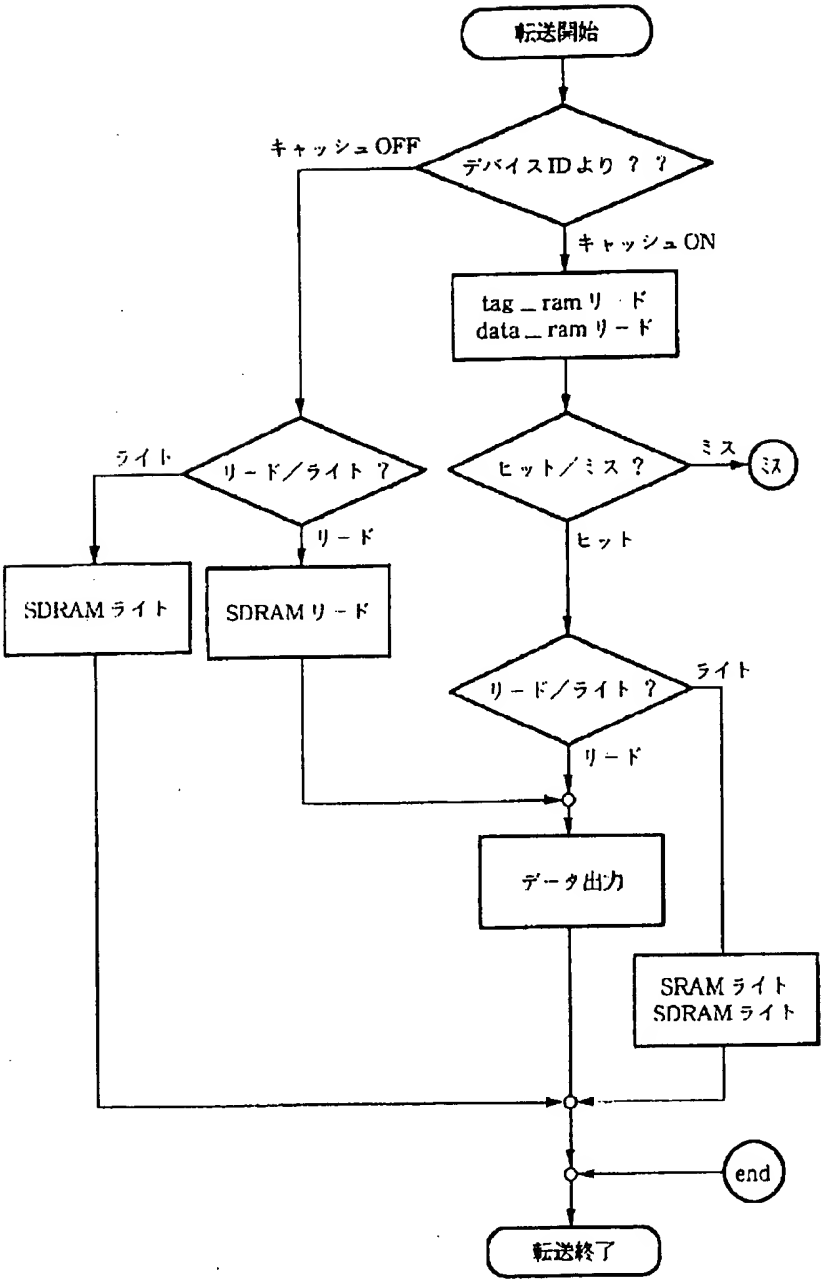


(クロックコントロール)
・BusのAgentは、Busのアクティビティーを検出し、クロックのOn/Offを自動で行なう。
・Sleep、WakeUp、Waitの3つのステートがある。
・SleepはBusAgentにアクティビティーがなく、クロックを停止させている状態である。
・Sleepの状態でも、デコーダー部、クロックコントロール部は動作しており、バスをモニタリング して、要求を待っている。
・デコーダーが自分のアドレスを検出すると、クロックゲートを開き、内部ロジックのクロックを動作させ、バスの要求に答える。ステートはWakeUpに移行する。
・データ転送が終了すると、Waitステートに移行し、次の要求を待つ。この時クロックは動作したままである。要求があれば、WakeUpステートに戻り、転送を行なう。また、要求を待っている間はタイマによりカウントを行ない、要求がないままカウンタがタイムアップした場合は、Sleepステートへ移り、クロックを停止させる。

【図56】


バースト転送開始
NO
バーストデータ<
キャッシュライン ?
YES
reg_ram リード
data_ram リード
ライト
リード/ライト ?
リード
ヒット/ミス ?
ミス
ミス
ヒット
SDRAM ライト
SDRAM リード
リード/ライト ?
ライト
リード
データ出力
SRAM ライト
SDRAM ライト
end
転送終了

【図58】



フロントページの続き

(72)発明者　横山　登
東京都大田区下丸子3丁目30番2号　キヤノン株式会社内

(72)発明者　加藤　勝則
東京都大田区下丸子3丁目30番2号　キヤノン株式会社内